Technische Universität Dresden
Fakultät Forst-, Geo-, Hydrowissenschaften
Institut für Kartographie

DIPLOMARBEIT

Kartographische Augmented Reality Anwendungen für mobile Geräte am Beispiel eines Campusführers der TU Dresden

Bearbeiter: Meike Viehweger

Betreuer: Prof. Dr.-Ing. Dirk Burghardt

Dipl.-Ing. (FH) Stefan Hahmann

Dresden, 14.03.2011

Selbstständigkeitserklärung

Hiermit erkläre ich, dass ich die von mir am heutigen Tage der Diplomkommission der Fachrichtung Geowissenschaften eingereichte Diplomarbeit zum Thema

"Kartographische Augmented Reality Anwendungen für mobile Geräte am Beispiel eines Campusführers für die TU Dresden"

vollkommen selbstständig und nur unter Benutzung der in der Arbeit angegebenen Literatur angefertigt habe.

Dresden, den 14.03.2011

_____

Unterschrift

Danksagung

Ich möchte mich ganz herzlich bei meinen Betreuern bedanken. Zum einen Prof. Dr.-Ing. Dirk Burghardt für die Themenfindung und die Unterstützung während der Erstellung der Arbeit, zum anderen Dipl.-Ing. (FH) Stefan Hahmann, der mir zu jeder Zeit mit Rat und Tat zur Seite stand und wesentlich zum Gelingen der Arbeit beigetragen hat.
Ebenfalls vielen Dank an die Firma Milan Geoservice GmbH für die Bereitstellung der Geländedaten.
Des Weiteren danke ich allen, die mir während meines Studiums und der Erstellung der Diplomarbeit fachlich und vor allem moralisch zur Seite standen.

Zusammenfassung

Die rasante Weiterentwicklung der Technik eröffnet vielen Lebens- und Wirtschaftsbereichen völlig neue Möglichkeiten. So ist die stetige Verbesserung von mobilen Geräten auch ein Gewinn für die Kartographie. Im Bereich der erweiterten Realität sind dazu schon einige Anwendungen entwickelt worden.

Diese Arbeit stellt verschiedene Augmented Reality Anwendungen vor, nicht nur aus dem Gebiet der Kartographie, sondern aus allen Lebensbereichen. Ein besonderes Augenmerk soll dabei auf der Anwendung mit mobilen Endgeräten liegen. Entstanden ist aus dieser Arbeit ein Campusführer, der nur die Namen der Gebäude anzeigt, welche der Nutzer von seiner Position aus auch tatsächlich sehen kann. Hierfür werden in der Arbeit Sichtbarkeitsanalysen im Allgemeinen und im Speziellen für GIS-Programme untersucht und vorgestellt. Auch die Beschriftung im dreidimensionalen Raum und auf dem Bildschirm von mobilen Geräten wird überblickshaft dargestellt. Abschließend wird der Campusführer getestet und bewertet sowie ein Fazit zum Thema Augmented Reality auf mobilen Endgeräten gegeben.

Abstract

Undreamed-of possibilities in many areas of life and also in different economic sectors emerge owing to the rapid enhancement of technology. The constant advancement of mobile devices is also a gain for cartography. In this field some augmented reality applications have already been developed. In this thesis some augmented reality applications, not only with cartographic references, are introduced. Special attention is paid to their use on mobile devices. Furthermore a campus-guide is developed, which only displays the points of interest actually seen from the user's position. For this purpose the concept of viewsheds is introduced and examined both in general terms and especially in the use of GIS-programs. The labeling in a three-dimensional scene and on the screen of mobile devices is shortly discussed as well. Moving on, the campus-guide is tested and evaluated. Also a conclusion on the topic of augmented reality with mobile devices is given.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

A-GPS	Assited Global Positioning System
ALS	Airborne Laserscanning
API	application programming interface
App	Applikation
AR	Augmented Reality
DGM	Digitales Geländemodell
DOM	Digitales Oberflächenmodell
ETRS89	European Terrestrial Reference System 1989
GIS	Geoinformationssystem
GPS	Global Positioning System
GSM	Global System for Mobile Communications
HMD	Head-Mounted Display
HUD	Head-Up Display
ITRS	International Terrestrial Reference System
JAX-RS	JavaAPI for RESTful Web Services
JSON	JavaScript Object Notation
LOS	Line of sight
OSM	Open Street Map
PDA	Personal Digital Assistent
POI	Point of Interest
REST	Representational State Transfer
SMS	Sort Message Service
SQL	Structured Query Language
TIN	Triangulated Irregular Network

UMTS	Universal Mobile Telecommunications System
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
UTM	Universal Transverse Mercator
WGS84	World Geodetic System 1984
WLAN	Wireless Local Area Network
XML	Extensible Markup Language

1 Einleitung

Anfang des Jahres 2011 feierte der Apple-Konzern zehn Milliarden Downloads in seinem App-Store. Diese Meldung war in vielen Zeitungen und Nachrichtensendungen zu vernehmen. Applikationen (Apps) für mobile Geräte, speziell aber für Smartphones sollen seit einigen Jahren unser Leben bereichern. So gibt es Anwendungen für jede Lebenslage. Touristen können sich Informationen zu Sehenswürdigkeiten anzeigen lassen, beim Straßenbau können die Leitungen, welche unter der Erde liegen, angesehen und somit ein Zerstören dieser verhindert werden. Aber auch einfache kleine Spiele, wie Basketball oder Fussball, können heruntergeladen werden. Dies kann natürlich nicht nur über den App-Store von Apple erfolgen, sondern auch über andere Anbieter von Apps für Smartphones. Insgesamt steigt der Markt und die Nachfrage nach diesen kleinen Programmen ständig an.
Neben vielen spielerischen Anwendungen können aber auch viele wirklich nützliche Programme, wie bereits angedeutet, genutzt werden. Auch in der Kartographie wurde sich an diesem Thema versucht und verschiedenste Applikationen sind dabei entstanden. In dieser Arbeit werden einige Beispiele im Bereich Kartographie, aber auch ganz allgemeine Anwendungen vorgestellt. Im Rahmen dessen wird auch der Begriff Mobilität und mobile Geräte geklärt sowie die Zugehörigkeit von verschiedenen Geräten zu diesen.
Ziel der Arbeit ist es, eine eigene Anwendung zu schaffen, welche die Namen der Gebäude der TU Dresden anzeigt, die vom Standort des Nutzers sichtbar sind. Auf Grund dessen wird im weiteren Verlauf der Arbeit die Berechnung von Sichtbarkeiten vorgestellt und die Durchführung in einem Geoinformationssystem (GIS) veranschaulicht. Außerdem wird überblicksmäßig erläutert, welche Möglichkeiten auf Seiten der 3D-Computergrafik gegeben sind, um dreidimensionale Szenen auf mobilen Endgeräten darstellen zu können.
Ein kurzer Blick wird auch auf die Beschriftung im dreidimensionalen Raum geworfen. Dabei wird besonders auf ihre Platzierung, Farbe, Art und Grad eingegangen.

Im Anschluss daran wird erläutert, wie der Campusführer entstanden ist und welche Berechnungen notwendig waren. Dabei werden Probleme benannt und Verbesserungen aufgezeigt. Außerdem wird die generelle Funktionsweise dieser Anwendung vorgestellt. Den Abschluss der Arbeit bildet ein Fazit sowie ein Ausblick zum Thema Augmented Reality auf mobilen Geräten.

2 Die erweiterte virtuelle Realität - Augmented Reality

Die erweiterte virtuelle Realität verbindet die Wirklichkeit mit computergenerierten Daten. Es erfolgt eine Ergänzung der realen Welt mit virtuellen Zusatzinformationen. Die reale Welt dient dabei als Basis, welche von virtuellen Objekten überlagert wird.
Erste Anwendungen der erweiterten virtuellen Realität finden sich bei der Überlagerung von Live-Übertragungen mit computergenerierten Grafiken, was vor allem für Sportbeiträge im Fernsehen genutzt wurde und wird. Es können z. B. die Bestmarken beim Skispringen eingeblendet oder bei der Auswertung von Fußballspielen die Laufwege der einzelnen Spieler dargestellt werden. Mittlerweile existieren zahlreiche Anwendungen in den unterschiedlichsten Lebensbereichen. So kann Augmented Reality (AR) in der Tourismusbranche eingesetzt werden, aber auch in der Medizin oder in Architekturbüros und auf Baustellen. Beispielsweise kann ein Architekt mit Hilfe eines mobilen Gerätes sofort sehen, wie sich sein gestaltetes Gebäude in die Landschaft einfügt. Touristen können sich Informationen zu Bauwerken anzeigen lassen, vor denen sie stehen. Ärzte können sich während einer Operation hilfreiche Informationen, wie Blutwerte oder Puls des Patienten, anzeigen lassen, ohne auf einen Monitor schauen zu müssen. Außerdem lassen sich mittlerweile 3D-Modelle der Organe anzeigen sowie geeignete Wege, um Instrumente einzuführen [Blümchen (2002)].

2.1 Definition der Augmented Reality

Nach Paul Milgram [Milgram und Kishino (1994)] gehört die erweiterte Realität zur gemischten Realität, welche sich zwischen virtueller und realer Welt einordnet. Die Einordnung der erweiterten Realität wird in Abb. 2.1 dargestellt.

Bevor die gemischte Realität definiert wird, soll erst einmal der Begriff der virtuellen und der realen Welt geklärt werden. Eine Definition der virtuellen Realität zu finden, ist sehr schwer. Alexander

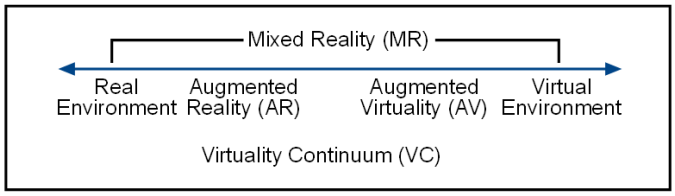


Abb. 2.1: Darstellung des „Realitäts-Virtualitäts-Kontinuum“ nach Milgram

Hennig [Hennig (1997)] definiert in seinem Buch „Die andere Wirklichkeit“ den Begriff virtuell als etwas nicht physikalisch Existentes. Das heißt, mit Hilfe eines Computers wird eine Welt simuliert, welche physikalisch in dieser Form nicht existiert. Realität definiert Hennig als etwas Nachweis- und Überprüfbares. Die Verbindung beider Begriffe, also die virtuelle Realität, definiert er wie folgt: „Virtual Reality ist eine Mensch-Maschine-Schnittstelle, die es erlaubt, eine computergenerierte Umwelt in Ansprache mehrerer Sinne als Realität wahrzunehmen“ [Hennig (1997)]. Im Umkehrschluss definiert sich die reale Welt als etwas physikalisch Vorhandenes, welche nicht durch einen Computer generiert wurde.

Auch Paul Milgram definiert die virtuelle Realität als eine synthetische, also künstlich erzeugte Welt. Diese virtuelle Realität kann sowohl Eigenschaften aus der realen Welt übernehmen, aber auch fiktive Elemente enthalten.

Die gemischte Realität vereint Elemente der virtuellen mit der realen Welt. Die erweiterte Realität ist ein Teil der gemischten Realität. Die reale Welt wird mit virtuellen Informationen ergänzt und kann somit einen Mehrwert für den Nutzer darstellen. Wichtig ist der Grad der Anreicherung mit computergenerierten Informationen. So liegt bei der Augmented Virtuality ein höherer Anteil an virtuellen Informationen als bei der Augmented Reality vor [Tegtmeier (2006)].

Eine erste Anwendung der erweiterten Realität stammt aus dem amerikanischen Militär. Dort wurden mit Hilfe von Head-Up Displays (HUDs) wichtige Flugdaten sowie ein Zielerfassungssystem in die Vorderscheibe der Flugzeugcockpits eingeblendet. Später wurde dieses System in Helme eingebunden [Hennig (1997)].

AR Anwendungen benötigen, vor allem zur Visualisierung, Hilfsmittel, welche sehr unterschiedlich sein können. Diese können z. B. HUDs bzw. Head-Mounted Displays (HMDs) sein. Mit Hilfe von HUDs werden die Informationen in das Sichtfeld des Nutzers projeziert. In Abb. 2.2 ist ein Beispiel für solch ein HUD angefügt. Ein HMD ist ein Ausgabegerät, welches am Kopf befestigt ist und com-

putergenerierte Bilder auf einen Bildschirm in Augennähe oder direkt auf die Netzhaut projiziert. Ein Beispiel dafür zeigt Abb. 2.3.

Abb. 2.2: Beispiel für eine Anwendung mit Hilfe eines HUD [IT-Wissen (b)]

Abb. 2.3: Beispiel für eine Anwendung mit Hilfe eines HMD [IT-Wissen (a)]

Weitere Visualisierungsmöglichkeiten können Projektoren sein, welche die reale Welt mit virtuellen, projizierten Elementen ergänzen. Aber auch Videoübertragungen, welche z. B. ein Sportereignis zeigen, können mit virtueller Information angereichert werden. Seit einiger Zeit finden sich auch bei mobilen Endgeräten viele AR Anwendungen. Auf diese wird in den nächsten Abschnitten noch genauer eingegangen.

Man kann Augmented Reality also als eine Verknüpfung von virtuellen und realen Informationen, welche eine Interaktion in Echtzeit erlaubt und die Synchronisation virtueller Objekte mit der realen Welt sicherstellt, beschreiben [Tegtmeier (2006)]. In der Dissertation von Tegtmeier findet sich resultierend aus den vorangegangenen Betrachtungen eine gute Definition, welche den Begriff AR umschreibt: „Augmented Reality beschreibt die Ergänzung der optischen und auditiven Sinneswahrnehmung durch die situationsgerechte Bereitstellung von computergenerierten Informationen“ [Tegtmeier (2006)].

2.1.1 Das Augmented Reality System

Ein AR System setzt sich aus drei wichtigen Komponenten zusammen. Als erstes muss die Position sowie die Orientierung des Aufnahmesystems bekannt sein. Wie genau diese Information sein muss, hängt sehr stark von der Anwendung ab. Prinzipiell gibt es zwei Methoden, um diese Parameter herauszufinden. Zum einen der positionsbasierte Ansatz, bei welchem Sensoren zum Einsatz kommen.

Zum anderen der musterbasierte Ansatz, welcher, wie der Name bereits sagt, Muster bekannter Objekte im Bild erkennt. Anschließend wird die virtuelle Szene berechnet, die Informationen werden abgerufen und an die Orientierung des Benutzers angepasst. Als letztes wird die Szene visualisiert. Dies kann, wie bereits erwähnt, durch verschiedene Systeme geschehen, wie z. B. HMDs, Smartphones oder mit Hilfe eines Projektors. Die drei Komponenten sind also die Positionsfindung, die Berechnung der virtiuellen Szene und Visualisierung dieser [Blankenbach (2007)].

Um AR Systeme verwirklichen zu können, benötigen die Geräte einen Videobereich. Zum einem, um z. B. mit einer Kamera ein Gebiet aufzunehmen, zum anderen, um am Ende der Anwendung die virtuellen Objekte auf einem Display visualisieren zu können. Es wird ein System zur Bildaufzeichnung und eines zur Bildausgabe benötigt. Dabei muss das bildaufnehmende System nicht zwingend nur die Kamera sein, es zählen auch Systeme zur Positions-und Orientierungsbestimmung zu ihnen. Um diese Parameter zu ermitteln, können Tracking-Verfahren genutzt werden. Mit Hilfe dessen können die Position und Orientierung des Nutzers bzw. des Displays und der Kamera, welche im Endgerät zumeist integriert sind, lokal bestimmt werden. Um die absolute Position des Gerätes zu ermitteln, wird in den meisten Fällen ein Satellitennavigationssystem, heutzutage fast ausschließlich das Globale Positionierungssystem (Global Positioning System (GPS)), verwendet. Doch um die Lage der Objekte, welche die Kamera aufnimmt, zu ermitteln, sind andere Methoden notwendig. Will man z. B. auf einen realen Tisch eine virtuelle Vase platzieren, muss die Position des Tisches im Raum bekannt sein, damit eine richtige Platzierung gewährleistet ist. Um dieses Problem lösen zu können, existieren im Wesentlichen zwei Tracking-Verfahren. Zum einen das sogenannte Natural-Feature-Tracking und zum anderen das Marker-Tracking.

Beim Natural-Feature-Tracking werden natürliche Objekte in der Umgebung erfasst. Dies erfordert einen sehr hohen Rechenaufwand, weshalb dieses Verfahren bei Anwendungen mit mobilen Endgeräten eher selten genutzt wird. Eine Möglichkeit dafür besteht über das Edge-Tracking, das heißt, es werden Ecken und Kanten aus einem 3D-Modell mit der realen Umgebung verglichen. Dies funktioniert am besten in einem stark bebauten Gebiet, da sich hier an Gebäudefassaden orientiert werden kann. Mit Hilfe von Transformationsmatrizen kann eine Berechnung der Orientierung der Kamera vorgenommen werden. Zu beachten ist die Kamerabewegung, welche festgestellt werden kann, wenn die Orientierung von mindestens zwei Positionen der Kamera berechnet wurde. Nähere Informationen zu diesem Thema finden sich in den Arbeiten von Reitmayer und Drummond [Reitmayr und Drummond (2006)] sowie von Tantius [Tantius (2008)].

Beim Marker-Tracking sind, wie es der Name sagt, künstlich angebrachte Marker notwendig. Auf dem Marker können Codezeichen angebracht werden, welche optisch ausgewertet werden können. Hierzu lassen sich ebenfalls weitere Informationen in der Arbeit von Tantius [Tantius (2008)] finden. Ein weiterer wichtiger Aspekt für AR Anwendungen ist die App an sich. Mit Hilfe der gewonnenen Information zur Position und Lage des Gerätes sowie der realen Gegenstände im Raum kann die Anwendung ausgeführt werden.
Nun muss ein Rendering, also die Erzeugung des virtuellen Bildes aus den Rohdaten erfolgen, um die virtuellen Informationen grafisch darstellen zu können. Dabei können sowohl zweidimensionale, als auch dreidimensionale Objekte visualisiert werden [Tantius (2008)].

2.2 Augmented Reality mit mobilen Geräten

Dieses Kapitel befasst sich zuerst mit den Begriffen Mobilität, mobile Geräte sowie mobile Dienste. Anschließend sollen einige Augmented Reality Beispiele mit mobilen Geräten aufgeführt werden.

2.2.1 Mobile Geräte und Dienste

Die Begriffe mobil und Mobilität tauchen in unserer Gesellschaft immer wieder und immer häufiger auf. Aber was heißt mobil eigentlich? Und was genau sind mobile Geräte? Mobil ist abgeleitet von dem lateinischen Wort mobilitas, was so viel bedeutet wie Beweglichkeit oder Schnelligkeit. Im Zusammenhang mit mobilen Geräten tauchen in der Fachliteratur verschiedene Begriffe, wie z. B. Mobile Computing, Nomadic Computing oder Ubiquitious Computing (Rechnerallgegenwart), auf [Frings (2002)]. Dabei bildet Ubiquitious Computing den Oberbegriff, welcher Nomadic Computing und Mobile Computing beinhaltet. Portable, also tragbare Geräte, welche aber, z. B. durch eine Netzwerkverbindung, doch räumlich begrenzt und somit an einen Ort gebunden sind, können in den Bereich des Nomadic Computings eingeordnet werden [Blankenbach (2007)]. Dies sind z. B. Laptops ohne UMTS-Hardware (UMTS: Universal Mobile Telecommunications System). Dabei handelt es sich um einen Mobilfunkstandard der dritten Generation. Dieser ermöglicht eine hohe Datenübertragungsrate, welche es erlaubt, nicht nur zu telefonieren, sondern auch im Internet zu surfen

[UMTS]. Laptops müssen zumeist um ein UMTS-Modul erweitert werden, da dieses standardmäßig nicht dazugehört, anders als bei den meisten Smartphones. Die Datenübertragung erfolgt über das Mobilfunknetz. Ist der Empfang zu diesem schlecht, verlangsamt sich also auch die Übertragungsrate. Global System for Mobile Communications (GSM) ist der Standard der zweiten Generation für Mobilfunknetze. Er dient hauptsächlich für die Telefonie sowie Kurznachrichten (Sort Message Service (SMS)), aber auch für Datenübertragungen [UMTS].
In den Bereich des Mobile Computing fallen Geräte, welche ebenfalls tragbar also ortsflexibel und über drahtlose Netzwerke miteinander verbunden sind. In diese Gruppe lassen sich Smartphones oder aber Personal Digital Assistents (PDAs) einordnen. Diese besitzen in der Regel UMTS-Module, um sich ortsunabhängig mit dem Internet zu verbinden [Blankenbach (2007)].

2.2.1.1 Mobile Dienste

Nach Reichwald [Reichwald (2002)] lassen sich die Eigenschaften mobiler Dienste in acht Spezifika zusammenfassen. Diese bilden zwei Kategorien, zum einen die Internet-, zum anderen die Mobilitäts-Spezifika.

- Internet-Spezifika
  - Digitalisierung und Automatisierung
  - Zeitflexibilität
  - Interaktivität, Integrativität, Vernetzung
  - Individualisierung
- Mobilitäts-Spezifika
  - Ortsflexibilität
  - Konnektivität
  - Personal Sphere
  - Umgebungssensibilität

Als erstes sollen die Internet-Spezifika betrachtet werden. Punkt eins ist die Digitalisierung und Automatisierung, das heißt, die Daten und Informationen werden automatisch in digitaler Form hergestellt, transportiert und gespeichert. Der nächste Punkt ist die Zeitflexibilität, welche beinhaltet, dass die Dienste jederzeit, rund um die Uhr, abgerufen und genutzt werden können. Wichtig sind auch die Interaktivität, Integrativität und die Vernetzung, da die verschiedenen Teilnehmer über Netzwerke (z. B. dem Internet) miteinander verbunden sind, Informationen austauschen und miteinander interagieren können. Außerdem erfolgt durch diese Vernetzung eine Integration des Nutzers in den Gesamtprozess, da er jederzeit auf die notwendigen Informationen zugreifen kann. Ein weiterer wichtiger Punkt ist die Individualisierung, welche die Grundlage der Personalisierung darstellt, da z. B. der Nutzer identifiziert werden kann und somit die Dienste optimal angepasst werden können.
Im Folgenden sollen nun die Mobilitäts-Spezifika betrachtet werden, zu denen als erster Punkt die Ortsflexibilität zählt. Da sich mobile Geräte drahtlos in Netzte einwählen und Informationen empfangen können, sind diese vom Aufenthaltsort unabhängig. Einschränkungen sollten allerdings gemacht werden, da z. B. nicht an jedem beliebigen Ort die Verbindung zum Internet ausreichend gewährleistet ist. Daraus folgt direkt der nächste Punkt, die ständige Konnektivität. Der Vorteil, ständig online zu sein, besteht darin, dass sich der Nutzer nicht jedesmal neu ins Internet einwählen muss. Hierdurch kann er ständig erreichbar sein und neuste Informationen sofort empfangen. Ein weiterer Punkt ist die Personal Sphere, das heißt, der Nutzer baut eine Art „persönliche Verbindung“ zu seinem mobilen Endgerät auf. Er passt es an seine Wünsche an und ist vertraut mit der Bedienung, dies fördert die Nutzung von mobilen Anwendungen. Als letzter Punkt soll die Umgebungssensibilität genannt werden, mit dessen Hilfe dem Benutzer, ensprechend der Informationen, welche die Erfassung und Auswertung von Umweltinformationen erbracht haben, Dienste und Anwendungen zu einem in diesem Moment passenden Kontext geliefert werden können [Blankenbach (2007)], [Reichwald (2002)].
Noch haben mobile Dienste und damit auch die mobilen Geräte an sich ihre Schwächen und erfüllen noch nicht alle Kriterien vollständig. So ist die ständige und genaue Positionsbestimmung ein Problem, welches noch nicht gelöst wurde (vgl. Abschnitt 3.1.2).

2.2.1.2 Mobile Geräte

Ein weiteres Schlagwort im Bereich mobile Geräte bzw. mobile Anwendungen ist das Wearable Computing. Dabei soll es um die Entwicklung eines tragbaren Computersystems gehen. In Zukunft sollen mehr und mehr elektronische Geräte im Miniaturformat in Kleidung, Schmuck oder Uhren eingebaut werden. Z. B. existieren Brillen, in deren Gestell ein kleiner Laser eingebaut wurde. Dieser Laser erzeugt ein Bild, welches auf ein Prisma im Brillenglas gelenkt wird. Mit Hilfe des Prismas wird das Bild auf die Netzhaut projiziert. Als Zukunftsvision ist für diesen Bereich etwa eine Brille mit integrierter Objekterkennung zu sehen, die dem Nutzer zu dem gesichteten Objekt dann die entsprechenden Informationen liefern kann [Mattern (2008)], [Amor (2002)].
Nach diesem kurzen Ausflug in die zukünftige Computerwelt soll im Folgenden auf mobile Endgeräte eingegangen werden. Kriterien von mobilen Endgeräten werden von Weiss [Weiss (2002)] aufgestellt, an welche sich nachfolgend orientiert wird:

- Sie arbeiten ohne Kabel, zumindest eine gewisse Zeit lang.
- Die Nutzung soll einfach und einhändig erfolgen, ohne das Gerät auf einer Unterlage abzustellen.
- Erweiterungen durch Apps und/oder einen Internetzugang müssen gewährleistet sein.

Nun stellen sich einige Fragen, welche gelöst werden müssen. Was ist ein Smartphone und ein PDA? Wo liegen Unterschiede und Gemeinsamkeiten? Sind Laptops auch als mobile Geräte anzusehen?
PDAs und Smartphones können unter dem Begriff Handhelds zusammengefasst werden. Dass heißt, es handelt sich um kompakte Geräte, welche mit einer Hand getragen und meist auch einhändig bedient werden können [Wagner (2007)].
PDAs sind kleine, tragbare Computer mit einem Betriebssystem, welche vor allem für die Termin- und Adressverwaltung genutzt werden. Die Tastatur befindet sich meist auf dem Display, welches als Touchscreen vorliegt. Mittlerweile wurden PDAs allerdings weitestgehend durch Smartphones abgelöst, da diese noch weitere Funktionen aufweisen. Eine Abgrenzung für diese Geräte zu finden ist schwierig, da mittlerweile sämtliche Handys nicht nur zum Telefonieren und SMS schreiben geeignet sind, sondern wesentlich mehr Fuktionen besitzen. Der Duden beschreibt das Smartphone als ein Handy, welches auch Adressen und Termine verwalten, sowie Fotos aufnehmen kann [Dudenredaktion (2006)]. Doch dies ist nicht weitreichend genug, da wie angedeuted diese Funktionen schon

einfache Handys efüllen. Dennoch sind diese nicht zwingend ein Smartphone. Eine geignete Definition für Smartphones zu finden ist nicht möglich, da es sich um ein Kunstwort handelt, welches von der Wirtschaft erfunden wurde. Prinzipiell kann man sagen, dass Smartphones die Funktionen von Handys und PDAs miteinander verknüpfen. So beinhalten sie neben den Kommunikationsfunktionen (telefonieren, SMS schreiben, E-Mail schreiben) einen MP3-Player, Terminkalender, aber auch einen GPS-Empfänger und weitere Sensoren sowie eine Kamera. Wichtig ist, dass sie ein eigenes Betriebssystem sowie einen relativ leistungsfähigen Prozessor aufweisen. Der Anwender kann sein Smartphone „aufrüsten“, indem er sich Apps aus dem Internet herunterlädt, je nachdem, welche Funktionen mit dem Smartphone erfüllt werden sollen. Die meisten Smartphones sind außerdem kleine Taschencomputer, mit denen man auch Textverarbeitungsprogramme ausführen oder PDF-Reader nutzen kann [IT-Lexikon].

Ein weiteres wichtiges Merkmal eines mobilen Endgerätes ist der geringe Stromverbrauch, welcher angestrebt wird. So ist eine Akkulaufzeit von mehreren Stunden bis hin zu Tagen sehr wichtig. Dies ist ein bedeutender Punkt für die Frage, wo ein Laptop eingeordnet werden kann. Sicherlich ist man mit einem Notebook ortsunabhängig und kann sich über ein drahtloses lokales Netzwerk (Wireless Local Area Network (WLAN)) an vielen Orten ins Internet einwählen. Dennoch sollte ein Laptop nicht als typisches mobiles Endgerät angesehen werden. So sind die Akkulaufzeiten doch nach wie vor sehr stark begrenzt (wenige Stunden). Manche Notebooks sind auch ziemlich schwer und somit nicht leicht von einem Ort zum anderen zu tragen. Noch schwieriger wird es, wenn ein Programm ausgeführt werden soll und sich der Nutzer mit dem Gerät weiterbewegen möchte. Netbooks hingegegen weisen schon wesentlich mehr Kriterien von mobilen Endgeräten auf. So sind sie wesentlich kleiner als Notebooks, allerdings auch leistungsschwächer als diese. Netbooks haben genau wie Laptops eine vollwärtige QWERTZ-Tastatur und ein Touchpad. Sie besitzen, im Gegensatz zu den meisten Smartphones, keinen Touchscreen. Die meisten Netbooks sind, genau wie Laptops, nicht UMTS-fähig. Sie können mit entsprechender Hardware (UMTS-Stick) nachgerüstet werden, um mobiles Internet zu ermöglichen. Durch ihre Größe lassen sich Netbooks einfach, auch im aufgeklappten Zustand, tragen und somit während des Gehens Programme und Anwendungen ausführen. Außerdem sind die Akkulaufzeiten im Vergleich zu Notebooks wesentlich besser, da die Hard- und Softwarekomponenten leistungsschwächer sind [IT-Lexikon].

Abb. 2.4 stellt die Größenunterschiede der beschriebenen Geräte dar. Dadurch wird verdeutlicht, dass Smartphones und Netbooks im Vergleich zu Laptops wesentlich mobiler sind.

Abb. 2.4: Größenunterschied zwischen Smartphones, Netbook und Laptop (v.l.n.r.) [thehotgadget (2010)]

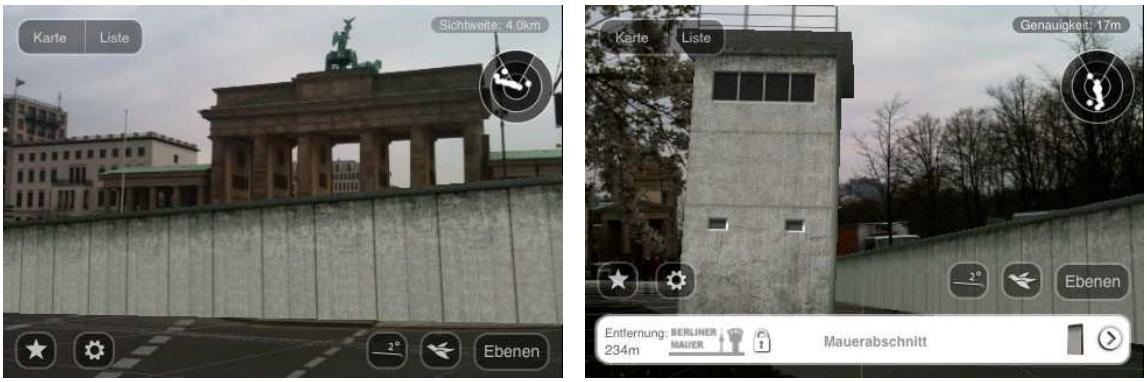
Auch Tablet-PCs, welche mit einem Stift oder den Fingern über den Bildschirm bedient werden können, stellen ein weiteres Beispiel für mobile Endgeräte dar. Sie weisen nur noch einen Bildschirm und keine Tastatur auf. Meist sind jene Geräte zudem mit einer Software zur Erkennung der Handschrift ausgestattet. Ursprünglich waren Tablet-PCs als eine Art elektronisches Notizbuch gedacht. Mittlerweile weisen sie aber annähernd den Funktionsumfang von Laptops auf. Durch ihre geringe Größe und ihr geringes Gewicht sowie durch die hohe Kompaktheit der Geräte können Tablet-PCs durchaus als mobile Endgeräte angesehen werden [IT-Lexikon]. Abb. 2.5 verdeutlicht den Größenunterschied zwischen einem Smartphone und einem Tablet-PC. Die Abmaße können natürlich auf Grund der verschiedenen Hersteller variieren. Tablet-PCs wurden erst seit Einführung des iPads im Frühjahr 2010 von den Konsumenten verstärkt angenommen, was zu einem erhöhten Angebot von Modellen anderer Anbieter auf dem Markt führte. Auf Grund der starken Konkurrenz zu Smartphones bzw. Laptops sind diese Geräte noch nicht weit verbreitet. Ungeachtet davon prognostizieren Marktforscher eine Verdreifachung der Verkäufe von Tablet-PCs bis zum Jahr 2014 [displaysearch]. Zusammenfassend kann festgestellt werden, dass eine Abgrenzung von mobilen Geräten nicht leicht ist. Notebooks und Netbooks können je nach Größe, Leistung und Stromverbrauch auch als mobile Endgeräte angesehen werden. Typische Beispiele sind jedoch das Smartphone und PDAs, welche die Eigenschaften der Mobilität im großen Maße erfüllen. Besonders der ständige Kontakt zum Internet ist noch nicht vollständig gewährleistet. Die Entwicklung von UMTS stellt einen großen Fortschritt dar, da mit Hilfe dessen das Surfen im Internet nahezu überall gewährleistet ist.

Abb. 2.5: Größenunterschied zwischen Smartphones und Tablet-PC [Computer-Bild]

Mobile Dienste müssen ebenfalls spezielle Kriterien erfüllen (vgl. Abschnitt 2.2.1.1), welche sich in Internet- und Mobilitäts-Spezifika gliedern. Diese können heutzutage auch noch nicht vollständig erfüllt werden, doch die Entwicklung schreitet mit großen Schritten voran. Nachfolgend sollen nun mobile Dienste im Bereich der Augmented Reality vorgestellt werden.

2.2.2 Mobile Anwendungen mit Augmented Reality

Ein erstes Beispiel stellt nicht nur eine mobile Anwendungen dar, sondern verdeutlicht auch die Einbindung von dreidimensionalen Objekten über die Layar-Plattform. Dieser AR Browser stellt einerseits Layer zum Download zur Verfügung, bietet aber auch die Möglichkeit der eigenen Anwendungserstellung (siehe 5.4). So kann beispielsweise über ihn ein Layer auf das Smartphone geladen werden, welcher die Berliner Mauer wieder auferstehen lässt. Befindet man sich am Brandenburger Tor in Berlin, kann man mit Hilfe dieser App den ehemaligen Verlauf der Mauer, welche 1989 gefallen ist, nachvollziehen. Auch ehemalige Wachtürme werden bei dieser Anwendung visualisiert [Layar (a)]. Die Abb. 2.6a und 2.6b geben einen Einblick in diese App.

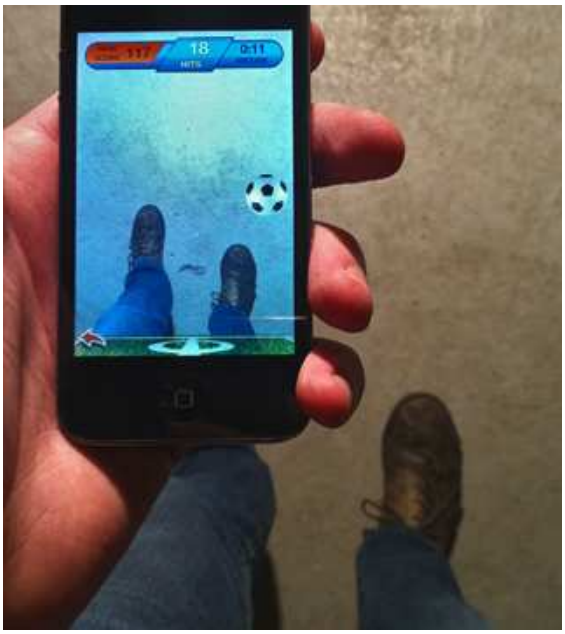


(a) Blick zum Brandenburger Tor mit Mauer im Vordergrund

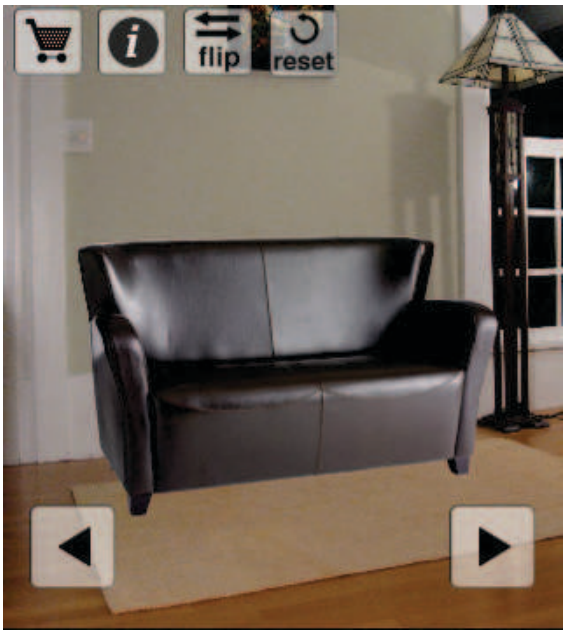
(b) Berliner Mauer mit Wachturm

Abb. 2.6: Layar-App Berliner Mauer, [Layar (a)]

Dieses Beispiel zeigt, dass es mittlerweile möglich ist, auch dreidimensionale Objekte mit Hilfe eines Smartphones zu visualisieren. Dies erfordert eine leistungsstarke Hardware des Gerätes.
Ein weiteres Beispiel mit dreidimensionalen Objekten ist die Einblendung des World Trade Centers in New York. Realtiv einfache dreidimensionale, auch texturierte Objekte, können also mittlerweile dargestellt werden. In Zukunft sollen auch komplexere 3D-Modelle eingebunden werden können. Mehr Informationen zu diesen Entwicklungen sind in Kapitel 6 der Arbeit aufgeführt.
Weitere AR Anwendungen für mobile Endgeräte existieren im Bereich der Spiele. So gibt es z. B. eine App zum Fussball spielen, in der der Spieler mit Hilfe des mobilen Gerätes einen virtuellen Ball sieht, welcher je nach realer Fußbewegung davonspringt. So kann der Nutzer einen virtuellen Ball von einem Fuß auf den anderen spielen, wie Abb. 2.7a zeigt [Apple (a)].
Ein drittes und letztes Beispiel für mobile AR Anwendungen soll eine App sein, welche bei der Einrichtung von Wohnungen hilft. Der gewünschte Gegenstand kann mit Hilfe dieser beliebig in der Wohnung platziert werden, da das mobile Gerät das Sofa, den Schrank oder Tisch virtuell im Raum erscheinen lässt. Abb. 2.7b vermittelt einen Eindruck, wie diese App funktioniert [Apple (b)].
Diese drei Beispiele zeigen, wie vielseitig AR auch auf mobilen Endgeräten sein kann. Zudem verbessert sich die Technik immer schneller, sodass mobile Endgeräte viel leistungsstärker und somit auch die Apps immer komplexer werden können.



(a) Fussball-Spiel [Apple (a)]



(b) Einrichtungshilfe [Apple (b)]

Abb. 2.7: Verschiedene AR Anwendungen

2.3 Augmented Reality in der Kartographie

Wie bereits in den voran gegangenen Abschnitten beschrieben, gibt es im Bereich der Augmented Reality sehr vielseitige Anwendungen. Auch im Rahmen der Kartographie lassen sich dabei zahlreiche Beispiele finden, auf welche im folgenden Kapitel genauer eingegangen werden soll.
Anwendungsbeispiele, welche allerdings nicht mobil sind, stellen die Augmented Maps dar, die von der Universität Cambridge entwickelt wurden. Eine gedruckte Papierkarte hat sehr viele Vorteile, neben topographischen Informtionen können sie zusätzlich auch sehr viele thematische Hinweise enthalten. Dank der Größe der Papierkarte erhält der Nutzer sehr schnell viele Überblicksinformationen. Bei digitalen Karten muss meist in die Karte hineingezoomt werden, um die Informationen richtig erkennen zu können. Ein Nachteil von gedruckten Karten besteht darin, dass keine bzw. nur recht eingeschränkt dynamische Prozesse dargestellt werden können. So ist unter anderem eine Visualisierung von Hochwasser- oder Feuerschäden in virtuellen Karten viel einfacher und besser zu realisieren. Aufgrund der Vorteile beider Karten liegt es nahe, analoge und digitale Karte miteinander zu verbinden. In dem Verfahren, welches an der Universität Cambridge entwickelt wurde, wird dafür eine Karte auf einen ebenen Grund, z. B. einen Tisch, gelegt. Eine Kamera beobachtet die Karte und

ihre Position wird berechnet. Ein Projektor fügt dynamische Prozesse hinzu. Somit können z. B. auf eine gedruckte Karte die überfluteten Flächen bei Hochwasser projiziert werden [Reitmayr (2006)], [Reitmayr u. a. (2006)]. In Abb. 2.8 wird die Funktionsweise und in den Abbildungen 2.9a sowie 2.9b ein Beispiel von Augmented Maps dargestellt.

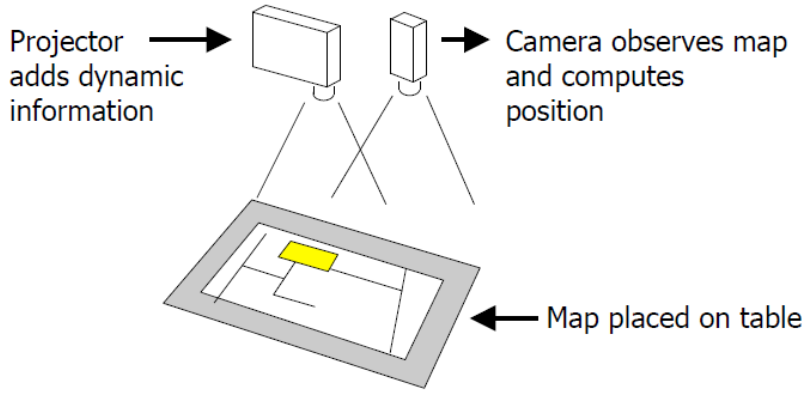
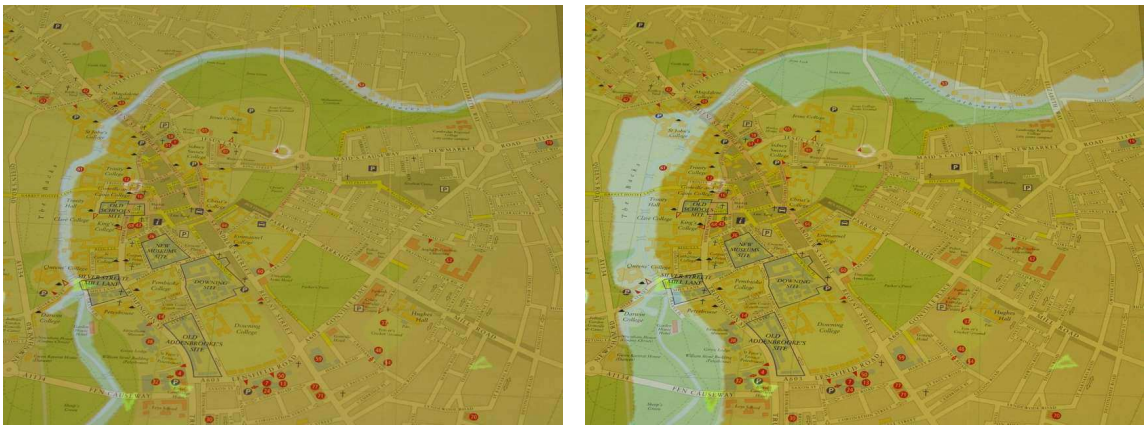


Abb. 2.8: Funktionsweise der Augmented Maps [Reitmayr (2006)]



(a) normaler Flusslauf auf Karte projiziert **(b)** Überflutung auf Karte projiziert

Abb. 2.9: Darstellung von Überflutungsflächen mit Augmented Maps [Reitmayr (2006)]

Ergänzend zu den Überflutungsflächen können z. B. auch Bilder von Sehenswürdigkeiten eingeblendet werden. Hierzu benötigt man ein weißes Kartonquadrat mit schwarzem Rahmen, welches am oberen Rand in der Mitte einen Zeiger vorweist. Dieser zeigt auf eine Position, welche die Kamera aufnimmt und somit einem Gebäude zuordnen kann. Jenes wird von einem Projektor auf dem Karton abgebildet (Abb. 2.10). Verschiebt man den Zeiger, verändert demnach seine Position, ändert sich auch das Bild, wenn von der entsprechenden Koordinate eines vorhanden ist.

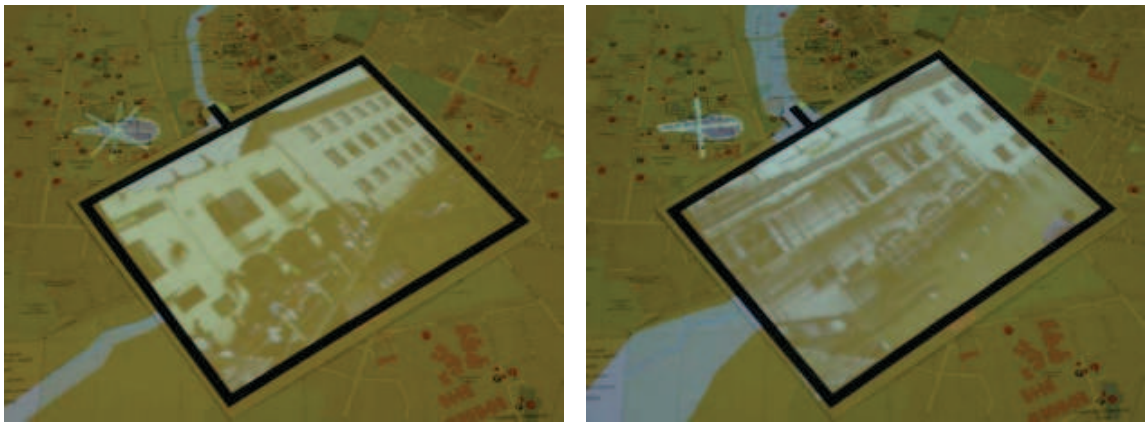


Abb. 2.10: Funktionsweise der Augmendet Maps mit Projektion einer Sehenswürdigkeit [Reitmayr (2006)]

Eine weitere Möglichkeit der Augmented Maps, zusätzliche Informationen zu erhalten, bietet die Hinzunahme eines PDAs. Mit Hilfe dessen können z. B. bestimmte Webseiten angezeigt werden, die zu der Position, auf welche das PDA zeigt, passen. So kann die Homepage eines Museums angezeigt werden, von welcher sich der Nutzer Informationen über Preise, Öffnungszeiten und Möglichkeiten zur Anfahrt mit dem öffentlichen Personennahverkehr einholen kann [Reitmayr u. a. (2006)].

Eine weitere kartographische Anwendung wurde an der ETH Zürich entwickelt und hat den Namen SwissPeaks. Dies ist nicht nur ein kartographisches Beispiel, sondern gleichzeitig auch eines für mobile Anwendungen. Die App soll Berggipfel erkennen, diese benennen und auf einem Smartphone visualisieren. Nun soll aber nicht jeder Berggipfel eines Gebirges angezeigt werden, sondern nur diese, die vom Standpunkt des Nutzers auch gesehen werden können. Das heißt, es muss eine Sichtbarkeitsanalyse (vgl. Kapitel 3) vorgenommen werden. Wichtig hierbei sind die Genauigkeiten verschiedener Komponenten des Smartphones. Dies ist zum einen die Genauigkeit des digitalen Kompasses, welcher durch Störungen, z. B. durch metallische Oberflächen, aber auch elektrische Ströme, sehr anfällig ist. Weiterhin schwankt die Genauigkeit des GPS-Empfängers, je nachdem, wie viele Satelliten verfügbar sind. Beeinflusst wird dies durch stark bebautes oder bewaldetes Gebiet, welches die Signale ablenkt. Außerdem muss der Öffnungswinkel der Kamera bekannt sein und somit das Sichtfeld. Nun muss natürlich auch die Leistung des Smartphones bedacht werden. So sollte das Display nicht mit zu vielen Informationen überladen werden, um lange Berechnungszeiten zu verhindern. Hierdurch werden zufriedenstellende Bildwiederholungsraten gewährleistet. Auf Grund dessen wurde die Anzahl der Berggipfel auf 60 begrenzt [Karpischek u. a. (2009)].

Da einige Punkte wegen ihrer Ungenauigkeiten bei der Positionsbestimmung nicht korrekt auf dem

Berggipfel liegen, umgibt die Signatur ein transparentes Band, welches je nach Fehlerbereich eine rote, orange oder grüne Färbung aufweist. Neben dieser Hilfestellung ist außerdem eine Funktion enthalten, welche es dem Nutzer erlaubt, diesen Fehler selbst zu beheben. Tippt man auf den Kompass auf dem Display, so gelangt man in einen Kalibriermodus, welcher die Behebung des Versatzes zwischen virtueller und realer Komponente erlaubt. Der fokussierte Berg erhält eine gelbe Punktfärbung, alle anderen sind blau dargestellt. Sollten mehr als 60 Gipfel für den Nutzer sichtbar sein, kann dieser auswählen, ob er nur die höchsten Berge oder aber jene, welche von seiner Position die kürzeste Entfernung vorweisen, visualisiert haben möchte. Außerdem können Informationen von der Online-Enzyklopädie Wikipedia [Wikipedia] hinzugeladen oder fehlende Gipfel über die Internetseite www.geonames.org hinzugefügt werden. Diese Plattform weist einen weltweiten geographischen Datenbestand auf, zeigt diese Daten in einer Karte an und erlaubt einen kostenlosen Download der Informationen [Karpischek u. a. (2009)].
Mittlerweile existiert neben der App SwissPeaks eine weitere mit Namen WorldPeaks, welche nicht nur die Gipfel der schweizer Berge benennen kann, sondern von Bergkuppen auf der ganzen Welt. Außerdem gibt es eine Anwendung namens peak.ar [Salzburg-research], welche ebenfalls die Beschriftung der Berggipfel vorsieht. Abb. 2.11 zeigt den Bildschirm eines Handys, welches die App SwissPeak aufgerufen hat.

Abb. 2.11: Die App SwissPeaks [Zürich]

Eine weitere wichtige AR Anwendung in der Kartographie ist die Navigation, welche im Folgenden durch das Beispiel der Fußgängernavigation näher erläutert wird. Auch diese App zählt zu den mobilen Anwendungen. Wichtig ist, wie schon im vorangegangen Beispiel, dass das Handy mit GPS

und einem elektronischen Kompass ausgestattet ist, außerdem wird ein Bewegungssensor benötigt. Weiterhin muss auch hier das Sichtfeld der Kamera bekannt sein, um die Informationen an dieses anpassen zu können. Die Kamera des Smartphones filmt die Umgebung und georeferenzierte Objekte, Straßen oder Point of Interests (POIs) werden mit der Lage des Smartphones verrechnet und auf dem Display abgebildet. Das Sichtfeld wird mit wichtigen Informationen der Route ergänzt. Diese wird mit Hilfe einer Liniensignatur visualisiert. Man kann sie sich wie eine Schnur vorstellen, der man folgen muss, um zum Ziel zu gelangen. Die Abbildungen in 2.12 zeigen die Visualisierungsart und lassen einen besseren Eindruck der Anwendung gewinnen. Nicht dargestellt werden, anders als die Nutzer es aus der Fahrzeugnavigation kennen, die Entfernung und Lage des Ziels sowie die Dauer, bis dieses erreicht wird. Somit wird der Nutzer nicht durch zu viele Informationen abgelenkt und kann sich voll und ganz auf die Route konzentrieren [Kluge (2010)]. Allerdings sind solche Informationen, gerade auch für ältere Nutzer, sicherlich von großer Bedeutung, da sie lange Wege nicht zu Fuß zurücklegen würden.

Abb. 2.12: 3D-Fußgängernavigation mit mobilen Geräten [Kluge]

Mario Kluge schreibt in seiner Arbeit aber auch von Problemen bei der exakten Überlagerung der virtuellen Szene mit der realen Umgebung. Perfekt wäre es, wenn die virtuellen Elemente in der Szene wirkten, als ob sie zur realen Welt gehörten. Bei aktuellen Anwendungen werden allerdings keine Licht- und Schattenverhältnisse sowie Oberflächenstrukturen berücksichtigt. Außerdem treten durch Schwankungen der Sensoren Ungenauigkeiten auf, sodass die virtuellen Ergänzungen nicht einwandfrei in die reale Welt eingegliedert werden können [Kluge (2010)].

2.4 Bewertung der vorgestellten Augmented Reality Anwendungen

Die vorangegangenen Abschnitte 2.2.2 und 2.3 haben gezeigt, wie vielseitig die mobilen AR Anwendungen sein können. Es gibt bereits zahlreiche verschiedene Apps auf dem Markt und es werden täglich mehr. Doch sind diese Apps wirklich sinnvoll und haben einen Nutzen?

Eine App zum Fußball spielen mit dem Smartphone hat eher weniger das Ziel, einen hohen Nutzen für den Anwender darzustellen. Es ist einfach ein Spiel, wie es normale Computerspiele auch sind und dient eher dem Zeitvertreib und Spaß. Einen höheren Sinn hat eine solche Anwendung sicherlich nicht.

Apps, welche 3D-Modelle von Möbeln oder sogar ganzen Häusern darstellen, sind sicherlich von Nutzen. So kann festgestellt werden, wie sich ein Gegenstand oder Gebäude in die Umgebung einfügt. Diese würden wahrscheinlich im Rahmen der Stadtplanung so manche Diskussion verkürzen bzw. erübrigen.

Informationen für Touristen, welche zu historischen Gebäuden eingeblendet werden, sind sicherlich für den Anwender sehr nützlich. Allerdings besteht dabei die Gefahr, dass man die reale Umgebung nur noch nebenbei wahrnimmt und sich sehr auf sein mobiles Endgerät konzentriert, was einer Kommunikation mit anderen Personen entgegenwirkt. Nutzt man z. B. die Möglichkeit, eine Stadtrundfahrt zu unternehmen, hat man auch die Möglichkeit, Zwischenfragen zu stellen, welche ein mobiles Gerät nicht unbedingt immer beantworten kann.

In der Kartographie kann die Augmented Reality sicherlich einige Vorteile erbringen. So ist die Fußgängernavigation mittels Smartphone eine sinnvolle Anwendung. Das vorgestellte Beispiel hat, wie bereits erwähnt, einige Nachteile, da beispielsweise die Distanz bis zum Endpunkt nicht angezeigt wird. Außerdem sollte man Navigationssysteme immer mit Vorsicht genießen, da die Informationsgrundlage nicht immer aktuell ist und eventuell einige Wege gar nicht oder fehlerhaft eingetragen sind. Das heißt, man sollte eine solche Anwendung als Unterstützung sehen, aber dennoch selbst auf die Umgebung achten und sich eventuell auf Übersichtplänen orientieren.

Zusammenfassend ist festzustellen, dass AR Anwendungen in vielen Fällen eine Bereicherung darstellen, aber dennoch nur als ein Hilfsmittel angesehen werden sollten, welches den Nutzer im täglichen Leben unterstützt. Man sollte mit gesundem Menschenverstand die Anwendungen nut-

zen und sich nicht komplett in diese vertiefen. Dies würde dazu führen, dass die Realität kaum noch wahr genommen wird und eine Kommunikation mit realen Personen immer mehr verringert wird.

3 Sichtbarkeitsanalyse

In diesem Kapitel sollen die Grundlagen zu Sichtbarkeitsanalysen (engl. viewshed) allgemein und im Besonderen für mobile Endgeräte vorgestellt werden. Sie berechnet alle Geländebereiche, welche von einem bestimmten Punkt aus sichtbar sind. Für mobile Endgeräte müssen einige Besonderheiten beachtet werden. So sollte z. B. die Datenmenge gering gehalten und die Displaygröße beachtet werden.
Sichtbarkeitsanalysen werden in sehr vielen unterschiedlichen Bereichen angewandt. So kann unter anderem geprüft werden, ob bestimmte Bauwerke das Gesamtbild einer Stadt stören und eventuelle Sichten blockieren. Außerdem kann bestimmt werden, was von einer Aussichtsplattform gesehen werden kann oder ob Funkmäste o. ä. auf eine Aussicht störend wirken [Roth (2009)].

3.1 Vorbetrachtungen zu ortsbasierten Sichtbarkeitsanalysen auf mobilen Endgeräten

In dieser Arbeit wird betrachtet, was von einem bestimmten Punkt aus gesehen werden kann und wie man in diesem Fall eine Sichtbarkeitsanalyse vornimmt.
Bevor mit einer Sichtbarkeitsanalyse begonnen werden kann, müssen einige grundlegende Dinge beachtet werden. So benötigt man z. B. gewisse Ausgangsdaten. Mit Hilfe eines Digitalen Geländemodells (DGM) werden die Höhen und Formen der Erdoberfläche wiedergegeben, allerdings ohne Vegetation und Kunstobjekte wie Gebäude, Brücken oder Leitungen. Für eine GIS-basierte Lösung muss das DGM als Rastermodell vorliegen. Je nachdem, welche Rasterweite dieses aufweist, ist auch die Genauigkeit des Standortes in diesem Bereich anzusiedeln. Das heißt, hat ein DGM eine Rasterweite von 25 m, dann kann auch der Standpunkt des Nutzers nur eine Genauigkeit von 25 m besitzen [Weigel (2007)]. Weiterhin muss die Augenhöhe des Betrachters beachtet werden. Laut Statistischen Bundesamt waren 2003 Männer in Deutschland durchschnittlich 1,77 m und Frauen 1,65 m

groß [Hahlen (2004)]. Wenn davon ausgegangen wird, dass die Augen ca. 10 cm unterhalb der Schädeldecke liegen, kann ein Durchschnittswert von 1,60 m für die Augenhöhe angenommen werden [Roth (2009)].

Wenn eine Sichtbarkeitsanalyse mit Gebäuden durchgeführt wird, muss außerdem beachten werden, dass die Vegetation eine wesentliche Rolle spielt. Z. B. kann ein Gebäude von einem großen Baum, aus einem bestimmten Winkel, nahezu bis vollständig bedeckt werden. Hierbei spielt auch die Jahreszeit eine Rolle. So ist die Bedeckung durch Vegetation im Winter geringer als im Frühjahr bzw. Sommer [Roth (2009)].

Weiterhin muss bedacht werden, dass der Mensch nur begrenzt weit blicken kann. Dies ist abhängig von der Auflösung der Netzhaut. Zwei Punkte müssen einen Mindestabstand zueinander haben, damit man diese gerade noch erkennen kann. Um ein Gebäude noch als solches definieren zu können, müssen bestimmte, markante Punkte noch erkannt werden, wie beispielweise die Haustür oder die Dachspitze. Die beiden Punkte haben einen gewissen Abstand zueinander und werden auf der Netzhaut abgebildet. Damit die zwei Punkte voneinander unterschieden werden können, müssen sie auf der Netzhaut von unterschiedlichen Rezeptoren angesprochen werden. Der Abstand der Punkte auf der Netzhaut zueinander muss größer sein als der Abstand zwei benachbarter Rezeptoren. Die Erkennbarkeit eines Gebäudes ist also durch die Abstände der Rezeptoren auf der Netzhaut gegeben, was der Auflösung des Auges entspricht. Bei durchschnittlichen Sichtverhältnissen sind zwei Punkte getrennt wahrnehmbar, wenn ihr Winkelabstand 2’ beträgt. Geht man von einer Höhe h des Hauses aus, welches sich in der Entfernung s befindet, so sieht man das Haus unter einem Winkel von $\tan\alpha = \frac{h}{s}$, wobei der Winkel α nicht kleiner als 2’ sein darf. Die Sichtweite kann durch atmosphärische Einflüsse verringert werden. So können z. B. Nebel, Schneefall und Regen die Sicht einschränken [Lindner u. a. (2006)].

Diese Aspekte müssen für Sichtbarkeitsanalysen im Allgemeinen beachtet werden. Im Beispiel des Campusführers ist die Sichtweite allerdings zu vernachlässigen, da in diesem Beispiel kein Gebäude solch eine große Entfernung aufweist, ohne dass sich davor eines befindet, welches dieses verdeckt.

Wie bereits erwähnt müssen bei Sichtraumanalysen mit mobilen Endgeräten zusätzliche Bedingungen beachtet und erfüllt werden. Wurde im vorangegangenen Abschnitt die durchschnittliche Sichtweite des Menschen behandelt, muss nun auch das Sichtfeld der Kamera mit einbezogen werden. Die Kamera, welche in den mobilen Endgeräten enthalten ist, hat nur ein bestimmtes maximales Aufnahmefeld. Dieses kann durch einfache Verhältnisgleichungen berechnet werden. Der Öffnungs-

winkel α der Kamera berechnet sich aus der Chipgröße b und der Brennweite c [Firchau (2001)]. Damit ergibt sich folgende Gleichung:

$$\tan \alpha = \frac{b}{c}$$

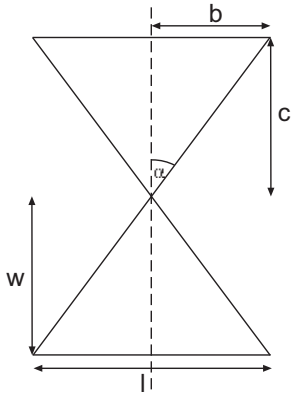


Abb. 3.1: Berechnung des Aufnahmefeldes einer Kamera nach [Firchau (2001)]

Wie in Abb. 3.1 zu erkennen ist, kann das Aufnahmefeld aber auch ohne die Chipgröße berechnet werden, denn diese ist nicht immer ausfindig zu machen. So ergibt sich eine weitere Gleichung zur Berechnung der Chipgröße mit Hilfe der Breite l und Entfernung w des Objektes:

$$b = \frac{(l \cdot c)}{(2 \cdot w)}$$

Mit Hilfe der berechneten Chipgröße kann anschließend der Öffnungswinkel der Kamera berechnet und somit das Aufnahmefeld bestimmt werden [Firchau (2001)].

Die Sichtbarkeitsanalyse auf mobilen Endgeräten hat besondere Anforderungen. Bevor es um diese gehen soll, muss als erstes geklärt werden, was ein ortsbasierter Dienst ist, was ihn ausmacht und wo er seinen Einsatz findet.

3.1.1 Ortsbasierte Dienste

Location Based Services, also standortbezogene Dienste, nutzen den aktuellen Standort eines Handys aus. Geographische Informationen, z. B. die Position des Nutzers, werden für Anwendungen verwendet, die einen Mehrwert für den Benutzer darstellen sollen. Schon die Suche nach einer Postleitzahl oder Adresse in einer Suchmaschine gilt als eine einfache Form von ortsbasierten Diensten. Dieser Ansatz ist natürlich sehr ungenau. Bei Handynutzern kann die Position viel exakter ermit-

telt werden, da man das Gerät meist bei sich trägt. Bei einfachen Handys kann der Standort durch ständigen Funkkontakt zu einer Basisstation festgestellt werden, solange das Handy eingeschaltet ist. Smartphones haben inzwischen eigene GPS-Empfänger, mit Hilfe derer sich die Position noch genauer bestimmen lässt [Schnabel (2010)].
Reichenbacher [Reichenbacher (2004)] unterscheidet bei Location Based Services zwischen Pull- und Push-Services. Pull-Services setzen einen aktiven Benutzer voraus. Der Anwender fragt seine Postion ab und wünscht sich in diesem Moment Informationen zu seinem Standort. Diese können dann unter anderem von einer Homepage abgefragt werden. Beispielsweise könnten Wikipedia-Artikel mit Informationen zur aktuellen Positionen erscheinen. Push-Services setzen einen passiven Nutzer voraus. Dem Anwender werden die Informationen förmlich aufgedrängt. So ist es vorstellbar, dass dem Handynutzer beim Vorbeigehen an einem Geschäft die neusten Angebote auf dem Smartphone angezeigt werden, obwohl er diese selbst gar nicht abgefragt hat [Merl (2003)].

3.1.2 Positionsbestimmung

Um ortsbasierte Dienste optimal einsetzen zu können, muss eine genaue Bestimmung der Position erfolgen. Dazu sind verschiedene Ansätze denkbar. Zum einen GPS sowie die Erweiterung dessen mit Hilfe des Mobilfunknetzes, das sogenannte Assited Global Positioning System (A-GPS). Aber auch die Ortsbestimmung mit Hilfe von WLAN ist möglich.

3.1.2.1 Global Positioning System

GPS wurde von den Vereinigten Staaten ursprünglich für militärische Zwecke entwickelt. Seit 1973 wurde an diesem Projekt gearbeitet und geforscht und erst im Jahre 1994 war das System voll funktionsfähig. Für außermilitärische Zwecke wurde GPS erst seit dem Jahr 2000 verstärkt verwendet, da ab Mai dieses Jahres der künstliche Fehler der Genauigkeit, welcher seitens des Militärs für nichtautorisierte Nutzer hinzugeschalten wurde, entfernt wurde. Dadurch können nun Postionsbestimmungen mit einer Genauigkeit von unter 10 m erreicht werden. Der GPS-Empfänger benötigt zu einer Positionsbestimmung eine simultane Messung zu vier Satelliten, da vier Unbekannte vorhanden sind, die Lage (zwei Unbekannte) und Höhe des Empfängers sowie der Empfängeruhrfehler. Genauere Informationen über GPS und dessen Funktionsweise finden sich in zahlreichen Fachbüchern wie z. B.

[Bauer (2003)] oder [Hofmann-Wellenhof u. a. (2001)].

Die Genauigkeit der Positionsbestimmung hängt sehr stark von der Umgebung ab, in welcher sich der Nutzer befindet. In stark bebauten Gegenden ist der Empfang der Signale durch den sogenannten Mehrwegeeffekt gestört. Die elekromagnetischen Wellen werden an den Fassaden der Gebäuden reflektiert und beeinflussen dadurch die Genauigkeit bei der Positionsbestimmung. Außerdem sind die Konstellationen der Satelliten nicht immer gleich. Sie bewegen sich auf ihren Bahnen um die Erde und benötigen für zwei Umrundungen ca. 23 h und 56 min. Das heißt, die Satellitenkonstellationen sind jeden Tag um ca. vier Minuten verschoben. Zu bestimmten Zeiten stehen für einige Orte die Satelliten nicht optimal, so dass die Genauigkeit der Positionsbestimmung darunter leidet [Wanninger (2005)].

3.1.2.2 Assisted GPS

A-GPS wurde speziell für den Einsatz in Handys entwickelt und kombiniert GPS mit dem GSM-Mobilfunknetz (vgl. Abschnitt 2.2.1). Durch die Verbindung zum Mobilfunknetz oder auch zum Internet wird das Handy schon grob positioniert und einer Funkzelle zugeordnet. Funkmasten in der Nähe messen die Signallaufzeiten und können dadurch eine ungefähre Position feststellen. Notwendig sind dazu mindestens drei Basisstationen. Ist diese ungefähre Position gefunden kann die Suche nach Satelliten beschleunigt werden, da über das Funknetz dem GPS-Empfänger Daten der Satellitenbahnen zugespielt werden.

Dieses System beschleunigt die Positionsbestimmung bei mobilen Geräten. Vor allem, wenn der GPS-Empfänger lange Zeit ausgeschaltet war, z. B., um Strom zu sparen. Auch eine Ortung innerhalb eines Gebäudes wird durchführbar, da die Verbindung zu den Funkzellen auch dort möglich ist.

3.1.2.3 Ortsbestimmung mit Hilfe von WLAN

Neben der Ortung mit Hilfe von Funk ist es auch möglich, mit Hilfe von WLAN die Position des Gerätes herauszufinden. Um dies zu ermöglichen, müssen die genauen Positionen der WLAN-Hotspots eingmessen sein. Diese Koordinaten werden auf einem Server abgespeichert. Die Systeme, welche sich hinter dieser Positionsbestimmung befinden, haben die Bezeichnung Wi-Fi Positioning System.

Die Entwickler dieser Systeme geben eine Genauigkeit von bis zu 5 m an [PC-Welt (b)]. Damit wäre die Technologie zumindest der einfachen GPS-Messung überlegen. Allerdings wird eine solche Genauigkeit sicherlich nur in Gebieten mit sehr vielen bekannten WLAN-Routern möglich sein, da somit eine genauere Berechnung der Position ermöglicht wird.

3.2 GIS-basierte Sichtbarkeitsanalyse

Führt man eine Sichtbarkeitsanalyse mit Hilfe eines GIS-Programmes durch, so basiert diese meist auf der Methode der Sichtlinie (Line of sight (LOS)). Bei dieser Methode wird untersucht, welche Punkte von einem gegebenen Standort aus in einem Gelände sichtbar sind. Zum besseren Verständnis soll diese Methode an einem Profil betrachtet werden. In diesem wird mit Hilfe einer Sichtlinie getestet, was von einem Punkt aus gesehen werden kann. Sobald die Linie das Gelände schneidet, ist alles dahinter liegende nicht mehr von der Ausgangsposition sichtbar. Sollte allerdings ein höherer Punkt nach dem ersten Schnittpunkt auftreten, wird dieser wieder mit der Sichtlinie geschnitten. Alles, was über diesem Schnittpunkt liegt, kann gesehen werden, alles, was darunter liegt, ist nicht sichtbar [Weibel (2009)]. Abb. 3.2 verdeutlicht die LOS-Methode.

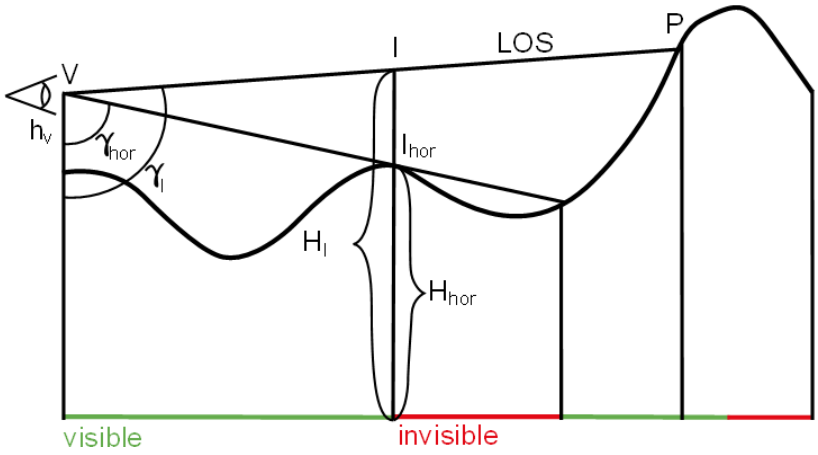


Abb. 3.2: Analyse der Sichtbarkeit mit Hilfe einer Sichtlinie nach [Trautwein u. a. (2010)]

Aus dem Bild zu entnehmen ist, dass geprüft wird, ob der Winkel zwischen der Senkrechten in V zur Strecke PV größer oder kleiner ist als zur Strecke des letztgespeicherten Punktes, in diesem Fall zu I [Trautwein u. a. (2010)].

Diese Berechnung der Sichtbarkeit wird als Binary Viewshed bezeichnet, da die sichtbaren Gebiete

den Wert 1 und die nichtsichtbaren Gebiete den Wert 0 erhalten. Doch reicht diese einfache Analyse wirklich aus? Man erhält verschiedene Sichtbarkeitskarten, wenn sich z. B. die Höhe des Beobachtungspunktes verändert oder aber ein Digitales Oberflächenmodell (DOM), welches Vegetation beinhaltet, gewählt wird. Für einen Punkt auf dem Gelände sind andere Objekte sichtbar, als für einen Punkt in 1,60 m Höhe. Genauso verändert sich die Sichtbarkeit, wenn plötzlich Bäume die freie Sicht behindern. Dies macht deutlich, dass die Wahl der Standorte und des Geländemodells die Analyse sehr stark beeinflussen [Trautwein u. a. (2010)].
Peter Fisher [Fisher (1992), Fisher (1993), Fisher (1996)] hat in zahlreichen Forschungen Sichtbarkeitsanalysen untersucht und ihre Anfälligkeit detailliert erforscht. Dabei wurden drei Bereiche genauer betrachtet:

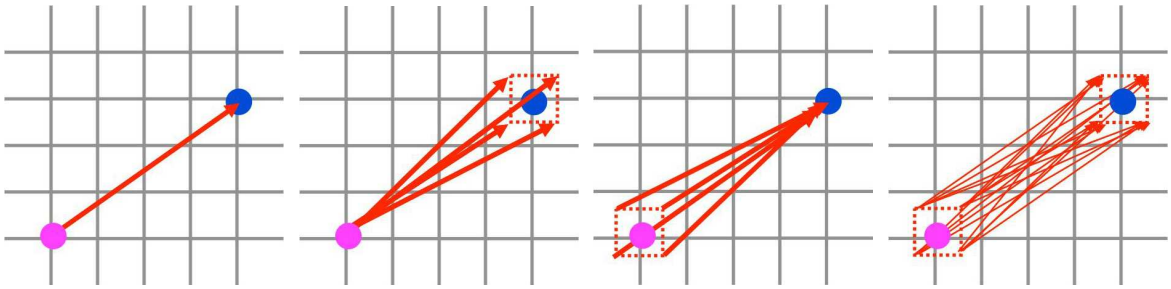
- Wie anfällig sind die Viewsheds in Hinsicht auf Höhenunsicherheiten im DGM?
- Wie unterscheiden sich die Ergebnisse bei unterschiedlichen Implementierungen der Algorithmen, welche zur Sichtbarkeitsanalyse verwendet werden?
- Ist dieses Grundprinzip der Sichtbarkeitsanalysen in unterschiedlichen Anwendungen sinnvoll? Sollte es erweitert werden? Oder sogar ein ganz anderer Ansatz genutzt werden?

3.2.1 Anfälligkeit von Viewsheds auf Höhenunsicherheiten im DGM

Digitale Geländemodelle beinhalten immer einige Fehler und Unsicherheiten. Diese können natürlich auch Auswirkungen auf die Sichtbarkeitsanalysen haben. Um dies zu erforschen, wurde ein zufälliger Fehler mehrmals in einem Geländemodell überlagert und somit ein Rauschen erzeugt. Danach wurden die Viewsheds der verrauschten Modelle mit den Originalen verglichen. Dabei wurde herausgefunden, dass die sichtbaren Bereiche im verrauschten Modell kleiner waren als bei den Berechnung mit den Originaldaten. Das heißt, es gab signifikante Unterschiede in den erstellten Viewsheds, was zu der Erkenntnis führt, dass es besser wäre, Wahrscheinlichkeiten der Sichtbarkeit für die einzelnen Geländepunkte zu berechnen und ihnen zuzuweisen. Diese werden als probable Viewsheds bezeichnet [Fisher (1992)], [Weibel (2009)].

3.2.2 Unterschiedliche Implementierung der Algorithmen

Welche Einflüsse haben nun die Algorithmen und ihre Implementierung auf die Sichtbarkeitsanalyse? Wie bereits verdeutlicht wurde, werden GIS-basierte Viewsheds mit Hilfe der LOS-Methode ermittelt. Es wird also getestet, ob eine Linie, welche vom Standort zu einem Beobachtungspunkt gezogen wird, das Gelände schneidet. Wird ein Schnittpunkt gefunden ist der Beobachtungspunkt unsichtbar, existiert keiner, ist er sichtbar. Dieses Grundprinzip gilt sowohl für Unregelmäßige Dreiecksnetze (Triangulated Irregular Network (TIN)) als auch für Rasterdaten. Es gibt verschiedene Arten, wie der recht einfache Ansatz implementiert werden kann. Die folgenden Betrachtungen gelten für ein Raster. Bei der Nutzung eines TINs können ähnliche Betrachtungen angestellt werden. Als erstes muss geklärt sein, ob es sich bei dem Beobachtungspunkt und den Testpunkten (Standorte) wirklich um einen Punkt oder eher um eine Zelle handelt. Es gibt dafür vier unterschiedliche Möglichkeiten. Zum einen könnte sowohl der Beobachtungspunkt als auch der Testpunkt ein Punkt sein, oder aber beide eine Zelle. Natürlich ist es auch denkbar, dass einer der beiden Punkte eine Zelle und der andere ein Punkt ist [Fisher (1993)]. Abb. 3.3 verdeutlicht die vier Varianten und ihre Unterschiede.



(a) Variante 1: von Punkt zu Punkt **(b)** Variante 2: von Punkt zu Zelle **(c)** Variante 3: von Zelle zu Punkt **(d)** Variante 4: von Zelle zu Zelle

Abb. 3.3: Verschiedene Möglichkeiten der Behandlung des Beobachtungs- und der Testpunkte [Weibel (2009)]

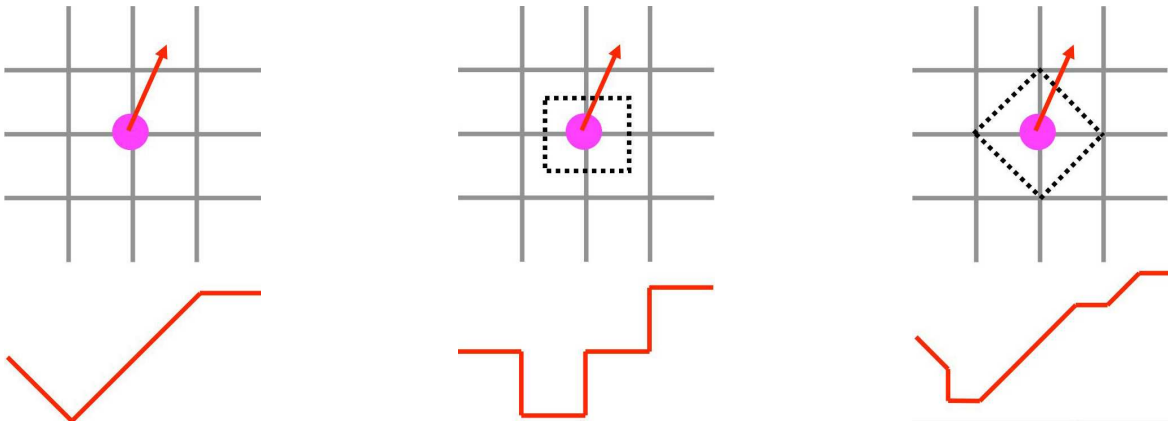
Wird von Punkt zu Punkt (Abb. 3.3a) getestet, entsteht eine Linie. Anders ist dies der Fall bei den Varianten zwei (Abb. 3.3b) und drei (Abb. 3.3c). Bei diesen entstehen vier verschiedene Linien. Variante vier (Abb. 3.3d) weist bereits 16 unterschiedliche Strecken auf. Bei der ersten Option entsteht die kleinste sichtbare Fläche, wohingegen die vierte Variante die größte Fläche liefert [Weibel (2009)].

Peter Fisher hat diese unterschiedlichen Varianten in zwei Testgebieten untersucht und ist auf große Schwankungen der Sichtbarkeiten gestoßen. Tabelle 3.1 zeigt die Unterschiede sehr deutlich.

Tab. 3.1: Vergleich der verschiedenen Methoden für die Behandlung der Punkte an zwei Testgebieten nach Peter Fisher [Fisher (1993)]

Beobachtungspunkt	Testgebiet 1 (Anzahl sichtbarer Zellen)	Testgebiet 2
Punkt zu Punkt (Variante 1)	2381	2034
Zelle zu Punkt (Variante 2)	3328	2271
Punkt zu Zelle (Variante 3)	2702	2666
Zelle zu Zelle (Variante 4)	3970	2907

Ein weiteres Problem ist die Höhenberechnung entlang der Sichtbarkeitslinie. Für diese gibt es drei verschiedene Möglichkeiten. Zum einen kann die Sichtlinie bei jedem Schnitt mit einer Rasterkante getestet werden. Die Höhe des Schnittpunktes wird aus den benachbarten Rasterpunkten interpoliert (Abb.3.4a). Eine weitere Möglichkeit bietet die Verdichtung der Maschen des Rasters mit Hilfe von Triangulationen. Hierbei wird ebenfalls ein Test der Sichtbarkeitslinie beim Schnitt einer Kante durchgeführt (Abb.3.4b). Eine dritte Option bietet die Betrachtung jeder Zelle als Prisma. Hierbei treten allerdings an den Kanten plötzliche Höhenänderungen auf (Abb.3.4c) [Fisher (1993)].



(a) Variante 1 **(b)** Variante 2 **(c)** Variante 3

Abb. 3.4: Verschiedene Möglichkeiten der Höhenberechnung entlang der Sichtbarkeitslinie [Weibel (2009)]

Die roten Profile sollen einen schematischen Eindruck der Höhen entlang der Sichtbarkeitslinien erzeugen. Tabelle 3.2 zeigt die Unterschiede zwischen den Methoden sehr deutlich auf. Besonders die dritte Methode liefert nur eine sehr kleine Sichtbarkeitskarte [Weibel (2009)].

Tab. 3.2: Bestimmung der Höhen entlang Sichtbarkeitslinien an zwei Testgebieten nach Peter Fisher [Fisher (1993)]

Höhenannäherung	Testgebiet 1 (Anzahl sichtbarer Zellen)	Testgebiet 2
Raster (Variante 1)	2381	2034
Triangulation (Variante 2)	3312	1917
Gestuft (Variante 3)	656	92

Ein weiteres Thema, mit welchem sich Peter Fisher beschäftigt hat, ist die relative Lage eines Punkte, also, ob er über oder unter einem anderen Punkt liegt. Diese Betrachtungen können mit Hilfe der Abb. 3.5 erklärt werden.

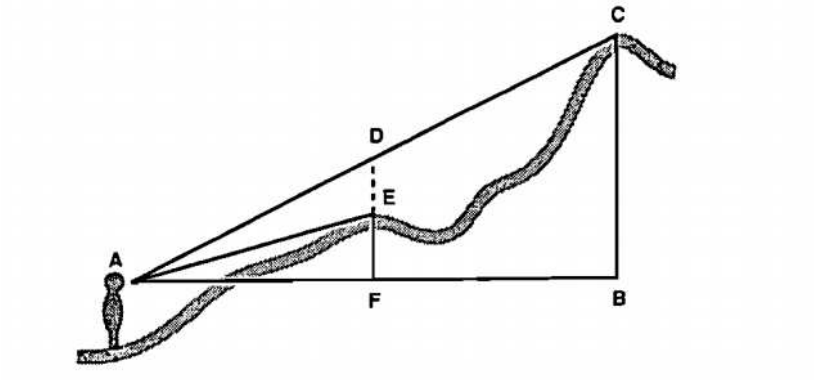


Abb. 3.5: Bestimmung der relativen Positionen [Fisher (1993)]

Es gibt verschiedene Bedingungen, welche erfüllt sein müssen, damit der Punkt C nicht vom Punkt E verdeckt wird. Ist eine dieser Kriterien nicht erfüllt, wird der Punkt C durch E verdeckt. Zusätzlich muss gelten, dass der Punkt C in jedem Fall höher liegt als A. Die Bedingungen lauten:

- AD > AE
- DF > EF
- $\angle ACB < \angle AEF$
- $\angle BAC > \angle FAE$
- AC/BC > AE/FE

Zur Berechnung dieser Tests sind verschiedene Ansätze denkbar. Eine Möglichkeit ist die Untersuchung der Höhe am Schnittpunkt der Sichtbarkeitslinie mit dem Raster. Eine zweite Variante ist der Vergleich der Gradienten der beiden Linien. Die dritte Möglichkeit ist eine Rundung aller Zwischenergebnisse auf 16 bit Integerwerte, also eine Integer-Arithmetik [Fisher (1993)]. Durch die unterschiedlichen Rechenansätze erhält man unterschiedliche Ergebnisse und damit auch verschiedene Sichtbarkeitskarten, wie in Tabelle 3.3 deutlich wird.

Tab. 3.3: Bestimmung der Sichtbarkeit eines Punktes durch Vergleiche an zwei Testgebieten nach Peter Fisher [Fisher (1993)]

Vergleich	Testgebiet 1 (Anzahl sichtbarer Zellen)	Testgebiet 2 (Anzahl sichtbarer Zellen)
Höhe (Möglichkeit 1)	2381	2034
Gradient (Möglichkeit 2)	2310	1877
Integer-Arithmetik (Möglichkeit 3)	2553	2052

Hier wird deutlich, dass diese unterschiedlichen Methoden einen nicht so großen Einfluss auf die Sichtbarkeit haben wie es in den beiden vorherigen Betrachtungen der Fall war.

3.2.3 Erweiterung von Sichtbarkeitsanalysen

Nun soll noch ein letzter Punkt betrachtet werden, nämlich die Erweiterung der Implementierung. Notwendig wird dies bei unterschiedlichen Anwendungsfällen. So ist der einfache Ansatz der LOS-Methode nicht immer sinnvoll. Peter Fisher hat in seiner Arbeit [Fisher (1996)] verschiedene Szenarien untersucht, so z. B. das Thema Waldbrand. Bei einem Feuer ist es wichtig, es frühzeitig zu registrieren. Ausgegangen wird beispielsweise von einem Turm, von welchem aus das Feuer entdeckt werden soll. Dabei ist es nicht zwingend erforderlich, den Boden und das Feuer direkt zu sehen, sondern es reicht aus, die Rauchentwicklung festzustellen. Dies kann natürlich viel schneller erfolgen, da Rauch bei einem Brand eher zu erkennen ist als die eigentlichen Flammen.
Ein anderes Beispiel wäre, wie sich ein neues Gebäude in die Umgebung einfügt. Hierbei ist wichtig, wie sich dieses zum Horizont verhält und ob es den Eindruck der Skyline, die Grenze der sichtbaren Erde mit dem Himmel, stört oder nicht. Abb. 3.6 verdeutlicht beide Situationen.

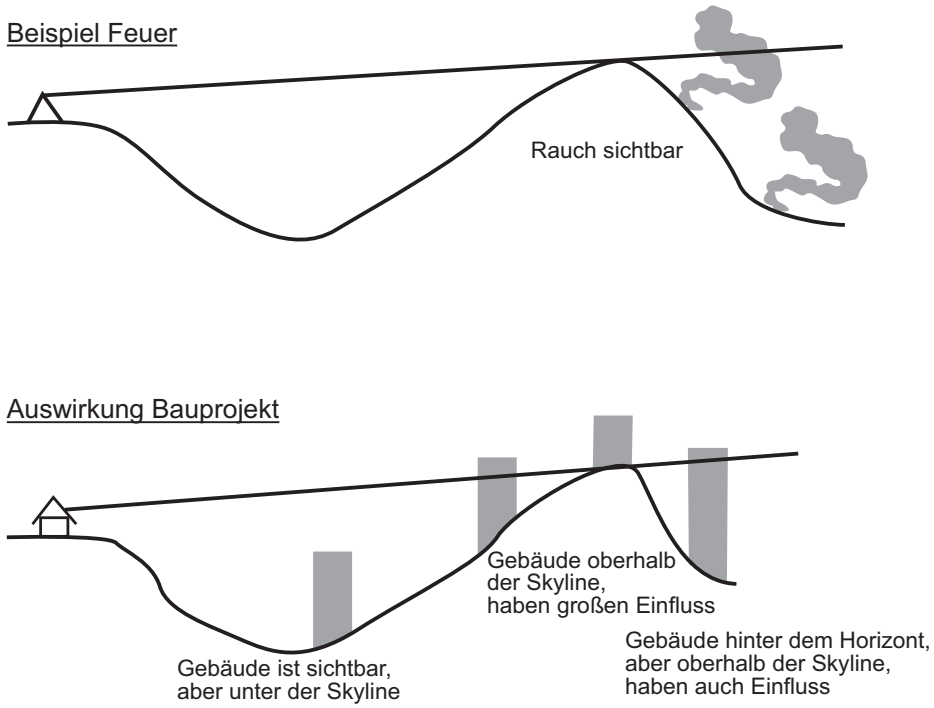


Abb. 3.6: Beispiele, bei denen eine binäre Sichtbarkeitsanalyse nicht geeignet ist nach [Fisher (1996)]

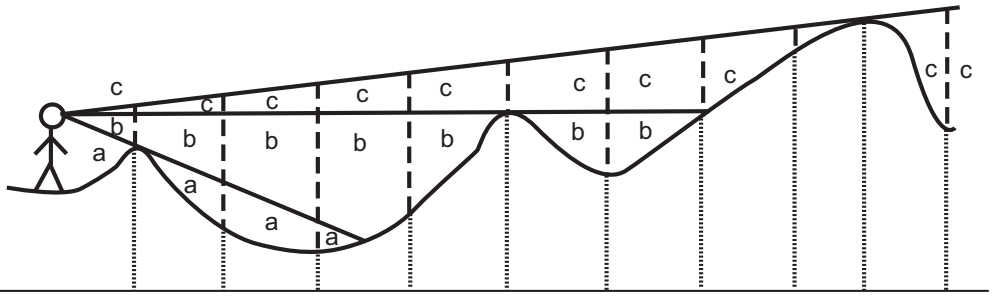
Um solche und ähnliche Situationen mit Sichtbarkeitsanalysen richtig berechnen zu können, bedarf es Alternativlösungen. Bevor diese betrachtet werden, soll noch einmal deutlich darauf hingewiesen werden, dass bei einer binären Sichtbarkeitsanalyse, welche bis jetzt ausschließlich betrachtet wurde, die Werte 1 und 0 zurückgegeben werden. Alle sichtbaren Bereiche erhalten den Wert 1 und alle nicht sichtbaren Gebiete den Wert 0.
Eine erste Alternative ist die Sichtbarkeitsanalyse mittels Horizonten (Horizons Viewshed). Dabei werden in speziellen Fällen folgende Werte zurückgegeben:

- 1: sichtbare Gebiete
- 2: lokaler Horizont (könnte z. B. eine Bergkuppe oder ein kleiner Hügel sein)
- 3: globaler Horizont (Skyline)
- 0: nicht sichtbare Bereiche

Weitere Alternativen wären Sichtbarkeitsanalysen mit lokalem oder globalem Offset, also Versatz (Local Offset Viewshed und Global Offset Viewshed). Wenn ein Gebiet sichtbar ist, wird der vertikale Abstand zum lokalen bzw. globalen Horizont abgespeichert. Ist das Gebiet nicht sichtbar, wird

der negative Versatz abgespeichert. Regionen, welche sich genau auf der Sichtlinie befinden, bekommen den Wert 0 zugewiesen [Fisher (1996)].

Abb. 3.7 zeigt die verschiedenen Möglichkeiten der Berechnung einer Sichtbarkeitsanalyse. Dabei wird deutlich, wie unterschiedlich die Varianten und ihre Rückgabewerte sind.



binäre Sichtbarkeitsanalyse	1	0	0	1	1	0	0	1	1	0
Sichtbarkeitsanalyse mittels Horizonten	2	0	0	1	2	0	0	1	3	0
Sichtbarkeitsanalyse mit lokalem Offset	0	-a	-a	b	0	-b	-b	c	0	-c
Sichtbarkeitsanalyse mit globalem Offset	b+c	a+b+c	a+b+c	b+c	c	b+c	b+c	c	0	-c

Abb. 3.7: Verschiedene Alternativen der Sichtbarkeitsanalyse nach [Fisher (1996)]

Jede dieser vier Methoden, kann auch rückwärts angewandt werden. Das heißt, es kann auch berechnet werden, von wo aus überall ein bestimmter Punkt gesehen werden kann [Fisher (1996)].

3.3 Sichtbarkeitsanalysen mit der Software ArcGIS

Die Software ArcGIS, welche zur Bearbeitung der gestellten Aufgabe genutzt wurde, erstellt eine binärische Sichtbarkeitskarte. Aus der Dokumentation [ESRI] geht hervor, dass die Berechnung von einem Punkt zu einer Zelle ausgeht. Das heißt, Variante 3 aus Abschnitt 3.2.2 (Abb. 3.3c) kommt zum Einsatz. Prinzipiell gibt es zwei verschiedene Möglichkeiten für die Berechnung in ArcGIS 10, zum einen das Werkzeug Sichtfeld, zum anderen Beobachterpunkte.

Mit Hilfe des Werkzeuges Sichtfeld wird ein Raster der Häufigkeit erstellt. Dieses gibt an, wie häufig ein Gebiet vom Standort bzw. von einer Polylinie von Standorten aus gesehen werden kann. Es ist also nicht nur möglich, Sichtbarkeiten von Punkten zu berechnen, sondern auch von Linien, wobei hier jeder Knoten bzw. Stützpunkt als Beobachtungspunkt genutzt wird. Der Häufigkeitswert, welcher bei der Berechnung des Sichtfeldes entsteht, wird im neuen Raster in der Attributtabelle

unter Value abgelegt. Werden Linien als Ausgangsdaten genutzt, so ist im Feld Value die Anzahl der Knoten erkenntlich, welche in jeder Zelle sichtbar sind. Hat der Beobachtungspunkt bzw. die Beobachtungspunkte keine eigene Höhe/Höhen, so wird diese durch bilineare Interpolation bestimmt. Liegt allerdings eine Zelle ohne Daten zum Beobachtungspunkt am nächsten, kann keine Berechnung der Höhe erfolgen und demzufolge auch keine Sichtbarkeit berechnet werden. Befinden sich ansonsten im Ausgangsraster Zellen ohne Werte, so werden sie in der Karte als nicht sichtbar dargestellt. Mit Hilfe des Werkzeuges Sichtfeld kann z. B. das Problem der Brandbeobachtung mittels eines Wachturms gelöst werden. Aber auch die beste Position für einen Freileitungsmast in einer Reihe kann berechnet werden.

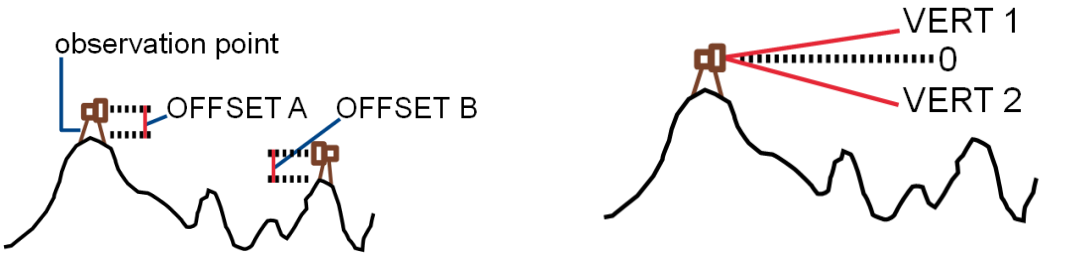
Das Werkzeug Beobachtungspunkte stellt eine binäre Sichtbarkeitsanalye dar. Dabei können auch mehrere Beobachtungsstandorte in den Ausgangsdaten vorliegen. Allerdings gibt es einen Grenzwert, welcher bei 16 Punkten liegt. Auch hier erhält der Beobachtungspunkt eine Höhe, welche durch bilineare Interpolation berechnet wurde, sofern noch keine eigene Höhe vorhanden ist.

Es besteht auch die Möglichkeit, eine Analyse der Sichtqualität durchzuführen, z. B. für alle Postionen auf der Erdoberfläche. Dazu müssen signifikante visuelle Merkmale ausgewählt und an diese ein Beobachtungspunkt gesetzt werden. Dies könnten beispielsweise Strommasten, Mülldeponien oder städtische Parks sein. Der Nutzer kann anschließend eine Gewichtung je nach Fragestellung vergeben. Möchte er ein Haus bauen, wirkt z.B. eine Mülldeponie im Ausblick störend. Somit wird für diese eine geringere Gewichtung vorgenommen. In Abhängigkeit der Sichtqualität und der Gewichtung, welche für die einzelnen Beobachtungspunkte vergeben werden kann, kann die Zellenposition mit den höchsten Punktwerten ausgewählt werden.

Es gibt verschiedene Möglichkeiten, die Sichtbarkeitsanalyse zu verändern, z. B. durch die Vergabe von Höhenwerten für die Beobachterpunkte oder vertikale Versätze. Insgesamt können neun verschiedene Parameter eingestellt werden. So kann eine Höhe des Betrachtungspunktes über SPOT eingestellt werden. Weiterhin kann ein Versatz genutzt werden. Dieser stellt die vertikale Entfernung von einem Punkt zur Geländeoberfläche dar. Hierbei unterscheidet man zwischen zwei verschiedenen Parametern. Einer, OFFSETA, legt die Höhe fest, welche dem Betrachtungspunkt hinzugefügt werden soll. Der andere, OFFSETB, legt fest, welche Höhe den Zellen im Raster zugefügt werden soll, die für die Sichtbarkeit Berücksichtigung finden sollen (Abb. 3.8a).

Ein weiterer Parameter ist der vertikale Winkel (Abb. 3.8b). Auch hier müssen ein Anfangs- und ein Endwert festgelegt werden, wobei der zweite Winkel kleiner als der erste sein soll. Die Winkel müs-

sen zwischen 90° und -90° liegen, positive befinden sich über der horizontalen Ebene und negative unter dieser. Die horizontale Ebene (0°) definiert sich aus der Summe der Höhe des Beobachtungspunktes und dem Wert von OFFSETA.



(a) Nutzung des Parameters OFFSET (Versatz) **(b)** Nutzung des Parameters VERT (vertikaler Winkel)

Abb. 3.8: Sichtbarkeitsanalysen mit ArcGIS [ESRI]

Mit Hilfe des Feldes AZIMUT (Abb. 3.9) kann ein Bereich für einen horizontalen Winkel für die Sichtbarkeitsanalyse angegeben werden. Dabei wird ein erster und ein zweiter Azimut festgelegt, wobei der Winkel zwischen beiden liegt und im Uhrzeigersinn verläuft. Erfolgt keine manuelle Eingabe, wird automatisch der Wert 0 angenommen. Der zweite zu vergebene Azimut muss auf jeden Fall größer sein als der erste. Wird vom Nutzer kein Endwert angegeben, wird standardmäßig von 360° ausgegangen.

Ein letzter Paramter ist der Radius (Abb. 3.9). Dieser definiert den Bereich, welcher bei der Identifizierung der von den Betrachtungspunkten sichtbaren Bereichen untersucht werden soll. Dabei werden die Zellen, welche eine zu große Entfernung aufweisen, nicht in der Analyse berücksichtigt. Es wird eine Startentfernung definiert, von welcher aus die Sichtbarkeit bestimmt wird. Zellen, welche Werte unter dieser Radiusangabe haben, werden als nicht sichtbar eingestuft. Allerdings kann dennoch eine Behinderung der Sichtbarkeit zwischen den beiden Radien vorkommen. Der Standardwert für den ersten Radius liegt bei 0. Zellen, welche außerhalb des zweiten Radius liegen, werden von der Sichtbarkeitsanalyse ausgeschlossen. Der Wert des zweiten Radius sollte immer größer sein als der des ersten. Hier wird als Standardwert „unendlich“ angenommen. ArcGIS interpretiert die Entfernungen zwischen den Radien als dreidimensionale Sichtlinien. Um eine richtige Berechnung der schrägen Strecke durchführen zu können, müssen die Einheiten des Geländes und der Z-Werte im selben Maßstab vorliegen. Damit die Radien als zweidimensionale planimetrische Entfernungen aufgefasst werden, müssen vor ihnen Minuszeichen gesetzt werden.

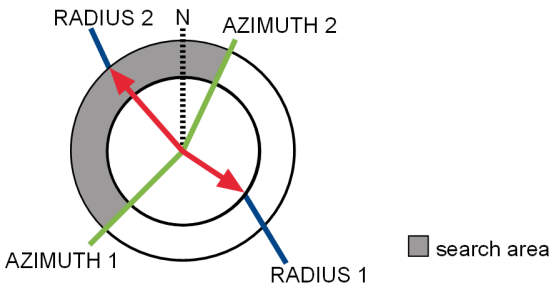


Abb. 3.9: Nutzung der Parameter Radius und Azimut bei Sichtbarkeitsanalysen mit ArcGIS [ESRI]

Neben diesen Parametern kann auch die Krümmung der Erde und die Brechung in der Atmosphäre korrigiert werden. Dazu müssen die Projektionsinformationen für die Oberfläche vorhanden sein. Außerdem müssen die Maßeinheiten in Fuß, Meter oder Einheiten pro Meter vorliegen [ESRI].

3.4 Grafische Darstellung der Sichtbarkeiten auf mobilen Geräten

Nachdem die GIS-basierte Sichtbarkeitsanalyse betrachtet wurde, soll nun ein Einblick in die Computergrafik gegeben werden. Um dreidimensionale Objekte anzeigen zu können bzw. Verdeckungen von Objekten zu beachten, müssen Berechnungen auf Seiten des Gerätes erfolgen. Die Leistungsfähigkeit von mobilen Endgeräten und die Geschwindigkeit der Datenverarbeitung ist noch recht eingeschränkt. Auch bei der Speicherung, Weiterverarbeitung und Darstellung von Daten treten bei diesen Geräten Defizite auf.

Die Grafikleistung der mobilen Geräte verbessert sich stetig. So strebt beispielsweise HTC, ein taiwanischer Hersteller von Smartphones und PDAs, eine Zusammenarbeit mit NVIDIA an, welcher einer der größten Entwickler von Grafikprozessoren ist [PC-Welt (a)].

In den folgenden Abschnitten sollen kurz verschiedene Verfahren für die grafische Darstellung auf mobilen Geräten vorgestellt werden. Roland Graf von der Universität in Salzburg hat sich intensiv mit diesem Thema beschäftigt [Graf (2003)]. Die prinzipiellen Verfahren sind auch in der Fachliteratur der 3D-Computergrafik zu finden [Watt (2002)].

3.4.1 Der Z-Buffer-Algorithmus

Als erstes soll der Z-Buffer-Algorithmus betrachtet werden. Mit dessen Hilfe können verdeckte Oberflächen entfernt werden. Notwendig für den Einsatz sind gerasterte Daten, von denen der Z-Buffer für jedes einzelne Pixel den Farbwert und zusätzlich einen Z-Wert der Bildtiefe des Pixels speichert. Über diesen Z-Wert, die Bildtiefe, wird dann die Sichtbarkeit des Pixels bestimmt. Sollen also zwei verschiedene Objekte in einem Pixel dargestellt werden, so werden die beiden Tiefenwerte der Grafikchips miteinander verglichen und es wird nur das Objekt dargestellt, welches zum Beobachter am nächsten liegt. Die Tiefeninformation des gewählten Pixels wird danach im Z-Buffer abgespeichert und der alte Wert wird durch diesen neuen ersetzt. Durch das beschriebene Verfahren können ferne Objekte durch nahegelegene verdeckt werden, wodurch ein natürlicher Tiefeneindruck entsteht [Graf (2003)].

Nahezu alle heutigen Grafiklösungen arbeiten mit dem Z-Buffer, wobei verschiedene Speichertiefen realisierbar sind. So können bei einem 8 Bit Z-Buffer Artefakte, das heißt, fehlerhafte Darstellungen, entstehen, welche bei 16 Bit und 32 Bit Speichertiefe seltener auftreten [Bauer].

Bei diesem Verfahren werden zwei zueinander nahe Objekte besser dargestellt als entfernt voneinander liegende, da die Werte der Abstände nicht gleichmäßig im Z-Buffer abgelegt und somit nahe Objekte genauer abgespeichert werden. Auch diese können zu Artefakten führen, in dem Fall, dass sie sich voneinander entfernen.

Um eine neue Szene darstellen zu können, muss der Z-Buffer gelöscht werden, wobei ihm ein einheitlicher Wert zugewiesen wird, der meist Null ist [Bauer].

Das Verfahren des Z-Buffers benötigt einen großen Teil des Speicherplatz des Rechners. Es gibt verschiedene Methoden, diesen Bedarf zu verringern, z. B. mit Hilfe von verlustfreien Kompressionen der Daten. Eine weitere Möglichkeit besteht in der Speicherung der Tiefeninformation mit alternierenden Vorzeichen, welche in den Z-Buffer geschrieben werden. Das heißt, es wird ein Bild mit positiven und das nächste mit negativen Vorzeichen gespeichert, was die vielen Löschvorgänge überflüssig macht [Graf (2003)].

3.4.2 Das Raytracing Verfahren (Strahl-Verfolgung)

Dieses Verfahren ähnelt dem der GIS-basierten Sichbarkeitsanalyse mit Hilfe von Sichtlinien. So kann man sich bei diesem Verfahren vorstellen ein Lichtstrahl würde berechnet werden, welcher den Beobachtungspunkt erzeugt. Der Algorithmus geht hierbei Pixelweise vor und berechnet für diese alle dargestellten Objekte sowie ihre Helligkeit. Jene können natürlich von anderen Objekten in der Umgebung des Pixels abhängen [Engels (1998)]. Somit müssen eventuell mehrere Helligkeiten berechnet werden. Diese Methode ist allerdings sehr rechenintensiv und daher weniger für mobile Endgeräte geeignet.

3.4.3 Verschiedene Culling-Verfahren

Culling-Verfahren haben das Ziel, möglichst früh nicht sichtbare Bereiche zu verwerfen, also nicht zu rendern und dadurch einen Gewinn an Performanz herbeizuführen. „Cull“ kommt aus dem Englischen und bedeutet soviel wie Auswahl oder Abfall.
Diese Verfahren können auch nacheinander angewendet werden. Marcel Lancelle bietet in seiner Diplomarbeit [Lancelle (2003/2004)] eine oft durchgeführte Reihenfolge dieser Verfahren an:

- Contribution Culling oder Detail Culling
- Frustum Culling
- Occlussion Culling
- Backface Culling

Als erstes werden mit Hilfe des Contribution Cullings Objekte entfernt, welche eine sehr kleine projizierende Fläche vorweisen. Danach wird das Frustum Culling angewendet, welches Objekte beseitigt, die sich außerhalb des Sichtbereiches der Kamera befinden. Verdeckte Objekte werden mit Hilfe des Occlussion Cullings gelöscht. Zum Schluss entfernt das Backface Culling die Rückseiten der Polygone.
Demnach werden viele Objekte bereits vor dem Rendering verworfen, was zu einer höheren Performanz führt und deshalb für mobile Geräte besonders geeignet ist.

4 Beschriftung im dreidimensionalen Raum

Dieses Kapitel beschäftigt sich mit der Beschriftung im dreidimensionalen Raum, welche bei mobilen Endgeräten ebenfalls entsprechende Anforderungen aufweist. Da die Plattform Layar (vgl. 5.4), welche für die praktische Arbeit genutzt wurde, im Bereich Beschriftung nicht besonders viele Möglichkeiten bietet, soll in diesem Kapitel ein kurzer theoretischer Überblick über Möglichkeiten der Beschriftung gegeben werden. Diesbezüglich wird nur eine Beschriftung von Gebäuden untersucht, da dies für den praktischen Teil der vorliegenden Arbeit von entscheidender Wichtigkeit ist.

4.1 Beschriftungsplatzierung

Beschriftungen sind an Objekte geknüpft [Maass und Döllner (2006)]. Mit Hilfe von Beschriftungstechniken kann eine optimale Lage und Ausdehnung der Schrift berechnet werden. Im dreidimensionalen Raum müssen besondere Aspekte beachtet werden, so z. B. die Verdeckungen durch Gebäude oder Bäume sowie die perspektivische Darstellung. Handelt es sich um eine interaktive Umgebung, müssen die Beschriftungen zudem dynamisch an die auftretenden Sichtverhältnisse angepasst werden.

Eine erste Beschriftungstechnik liefern Maass und Döllner [Maass und Döllner (2006)]. Dabei wird die Schrift komplanar auf einer Objektebene angebracht. Hierzu sollte beachtet werden, dass die Beschriftung eindeutig ist sowie mögliche Verdeckungen vermieden werden. Abb. 4.1 verdeutlicht mögliche Fehler. So sind die Beschriftungen von „School“, „Cathedral“ und „Library“ nicht eindeutig zu den Gebäuden zuzuordnen. Des Weiteren ist nicht erkennbar, ob es sich bei der Schule um das verdeckte Haus im Hintergrund oder jenes im Vordergrund handelt, bzw., ob es überhaupt zwei einzelne Gebäude sind. Von diesem Blickwinkel aus betrachtet wirken die beiden Häuser wie ein Gebäude [Lehmann und Döllner (2010)].

Abb. 4.1: 3D-Beschriftung nach der Technik vom Maass und Döllner [Lehmann und Döllner (2010)]

Das hier dargestellte Verfahren basiert auf den sogenannten „eingebetten Annotationen“, das heißt, die Beschriftung wird auf einer Ebene angeführt, welche parallel orientiert zum zu beschriftenden Objekt liegt. Die gewählte Fläche im Objekt soll jene mit der geringsten Distanz zum Betrachter sein. Mit Hilfe dieses Verfahrens soll eine optimale Beschriftungsplatzierung im dreidimensionalen Raum ermöglicht werden, was aber, wie bereits angeführt, nicht in jedem Fall gelingt [Lehmann und Döllner (2010)].

Verbessert werden kann dieses Verfahren durch die Ermittlung der sichtbaren Bereiche des Bildes. Dies geschieht bildbasiert, indem die Bildpunkte einzeln auf Sichtbarkeit untersucht werden. Mit Hilfe der Bildschirmkoordinaten der sichtbaren Bereiche können diese Ebenen als 2D-Polygone extrahiert werden. Die Unterscheidung der Gebäude wird durch die Vergabe von Referenznummern an die sichtbaren Bildpunkte ermöglicht. Dabei wird der größten sichtbaren Ebene eines Objektes die Beschriftung zugeordnet. Diese richtet sich am Mittelpunkt der Ebene aus. Es erfolgt eine Transformation vom Bildraum in den Objektraum und eine Zuweisung der Beschriftungsgeometrien mit ihrer Position im dreidimensionalen Raum.

Abb. 4.2 stellt das Ergebnis dieser Erweiterung des ersten Verfahrens dar. Eine Verbesserung der Schriftplatzierung ist deutlich zu erkennen. So sind die Beschriftungen eindeutig zugeordnet und es liegen keine Verdeckungen vor. Für die Beschriftung der „Townhall“ hat sich die Platzierung nicht verändert, da sie bereits mit dem ersten Verfahren gut angeordnet war [Lehmann und Döllner (2010)].

Abb. 4.2: 3D-Beschriftung mit verbessertem Verfahren [Lehmann und Döllner (2010)]

In den Arbeiten von Thomas Kolbe [Kolbe u. a. (2004)] und Daniela Schulz [Schulz (2004)] werden weitere Methoden zur Beschriftung im dreidimensionalen Raum vorgestellt. Neben der komplanaren Beschriftung, welche schon vorgestellt wurde, wird die Methode der Schildbeschriftung erläutert. Dabei wird die Beschriftung rechtwinklig zum Gebäude platziert, wie in Abb. 4.3 deutlich wird. Die Beschriftung ist frontal zum Nutzer ausgerichtet, wodurch das Problem der perspektischen Verzerrung vermieden wird. Bei diesem Verfahren ist sowohl eine waagerechte als auch eine senkrechte Beschriftung denkbar. Durch die waagerechte Beschriftung wird die Verdeckung des zu beschriftenden Gebäudes gegenüber der senkrechten Beschriftung stark verringert, jedoch werden dadurch mehr Objekte im Hintergrund verdeckt.

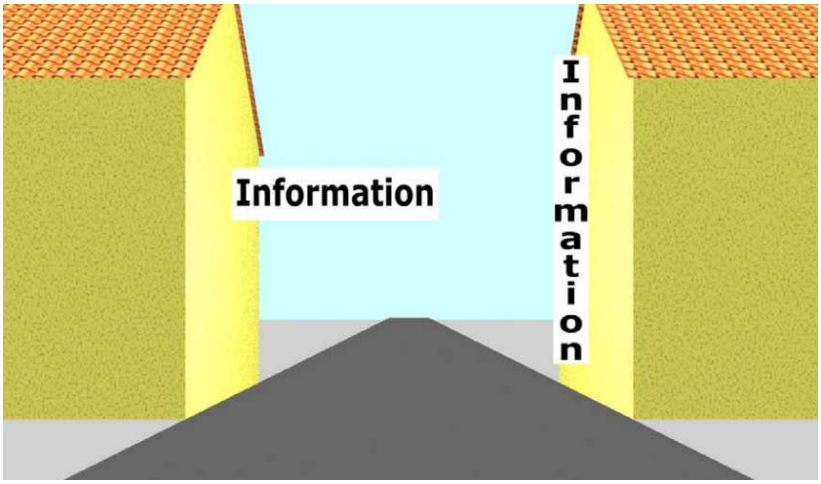


Abb. 4.3: Methode der Schildbeschriftung [Kolbe u. a. (2004)]

Die Abstandsbeschriftung ist der Schildbeschriftung sehr ähnlich, allerdings wird hier die Schrift, wie der Name schon sagt, in einem gewissen Abstand zum Gebäude angebracht. Um die Zuordnung zu gewährleisten, sollte ein Verbindungselement, beispielsweise ein Pfeil, visualisiert werden. Diese Form der Beschriftung benötigt sehr viel Platz und sollte deshalb weitestgehend vermieden werden. Geeignet ist diese Methode besonders bei der Beschriftung von kleinen Elementen, wie z. B. von weit entfernten Gebäuden oder kleinen Denkmälern. Abb. 4.4 verdeutlicht die Methode der Abstandsbeschriftung.

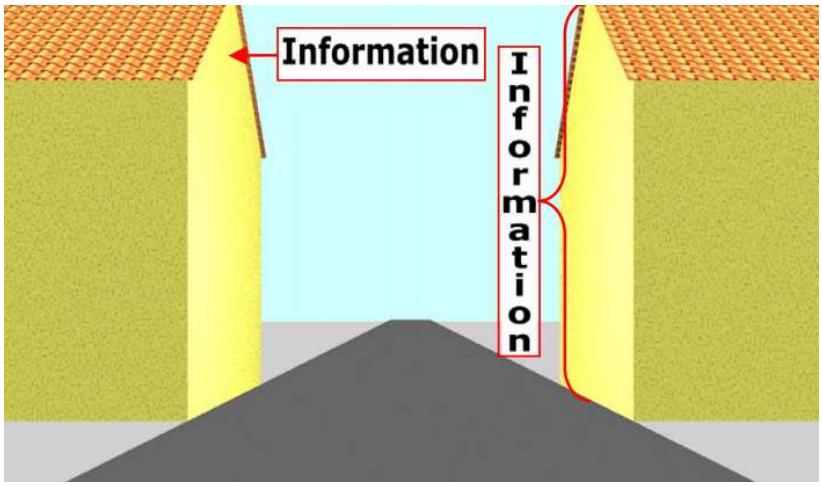


Abb. 4.4: Methode der Abstandsbeschriftung [Kolbe u. a. (2004)]

Mit Hilfe einer Klammer, welche als verbindendes Element genutzt werden kann, wird neben der Beschriftungszuordnung auch die Ausdehnung des Gebäudes verdeutlicht.

4.2 Schriftformen

Nachdem eine geeignete Platzierung der Schrift gefunden wurde, können nun die verschiedenen Formen einer Schrift betrachtet werden. Welche Schriftart und welche Größe sollte verwendet werden? Sind bestimmte Farben besser geeignet als andere? Ist eine Freistellung der Schrift sinnvoll? Diese und andere Fragen werden im Folgenden betrachtet.

4.2.1 Schriftart

Die Schriftart stellt die charakteristischen Grundformen der jeweiligen Buchstaben dar. Schriftarten lassen sich in Schriftgattungen einordnen, welche den Schriftschnitt variieren. Der Schriftschnitt beinhaltet die Merkmale Schriftlage, Schriftbreite und Schriftstärke. Als Beispiel einer Schriftgattung kann die Groteskschrift genannt werden, welche sich dadurch auszeichnet, dass sie eine nahezu konstante Strichbreite und keine Serifen aufweist. Bei mobilen Geräten sollte darauf geachtet werden, dass möglichst eine serifenlose Schrift verwendet wird, da Serifen die Lesbarkeit eher beeinträchtigen, vor allem, wenn eine kleine Schriftgröße gewählt wird. Dies ist bei mobilen Anwendungen meist der Fall, da das Display nur eine beschränkte Größe aufweist und mit der Schrift möglichst wenig vom eigentlichen Bild verdeckt werden soll [Großer (2002)].

Die Schriftlage sollte auf jeden Fall gerade und nicht nach rechts- oder linksneigend gewählt werden, da sonst am Bildschirm der sogenannte Aliasing-Effekt auftritt. Das heißt, bei der Nutzung von einer kursiven Schrift treten Stufen an den schrägen Linien auf, welche auf dem Bildschirm zu verschwommenen Schriften führt, die nur schwer lesbar sind [Brunner (2001)], [Brunner (2002)]. Abb. 4.5 und 4.6 verdeutlicht das Problem des Aliasing-Effektes und zeigen, dass kursive Schriften definitiv vermieden werden sollten.

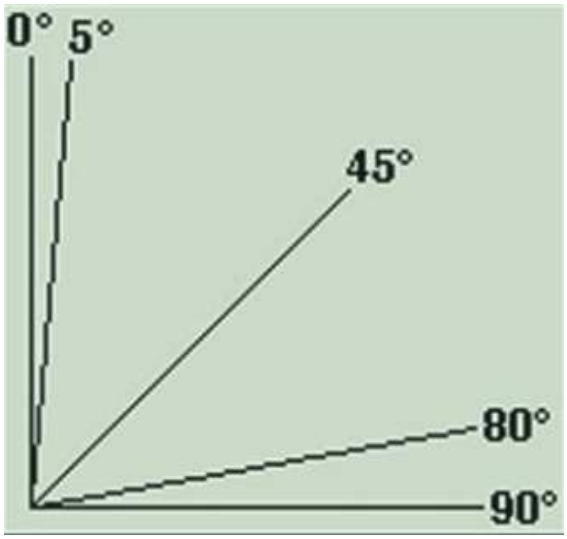


Abb. 4.5: Abhängigkeit der Richtung bei der Anzeige von Linien am Bildschirm [Brunner (2002)]

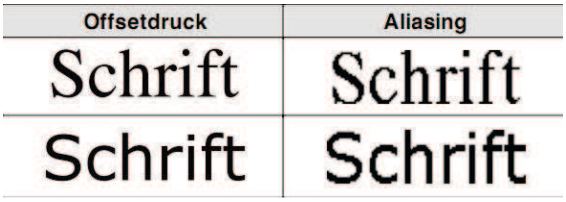


Abb. 4.6: Vergleich Schrift aus dem Offset-Druck mit Schrift am Bildschirm [Brunner (2001)]

Es wird deutlich, dass Linien, welche exakt senkrecht und waagerecht in der Bildpunktmatrix verlaufen keine Störungen aufweisen. Bei der Linie im 45°-Winkel sind die Störungen nur sehr gering, die Linie wirkt schmaler als die senkrechten und waagerechten Linien. Die größte Störung zeigen die Linien in der Nähe der Vertikalen und Horizontalen [Brunner (2002)].
Generell gilt, dass so wenig wie möglich verschiedene Schriftarten verwendet werden sollten. Dadurch wirkt das Gesamtbild harmonischer und eine Zuordnung der Schriften zu bestimmten Objekten wird für den Nutzer erleichtert [Großer (2002)].

4.2.2 Schriftfarbe

Mit Hilfe der Variation der Schriftfarbe können Objekte qualitativ unterschieden werden. Aber auch hier ist, ähnlich wie bei den Schriftarten, darauf zu achten, sich auf wenige Farben zu beschränken, da zu viele Schriftfarben das Bild unübersichtlich wirken lassen. Schwarz hat einen hohen Kontrast zu allen anderen Farben und ist somit als Schriftfarbe sehr geeignet. Eine Freistellung oder Umrahmung der Schrift erhöht die Lesbarkeit zusätzlich, da die Beschriftung durchaus auch auf dunklem Hintergrund auftreten kann. So kann eine weiße Umrahmung gewählt werden, um die schwarze Schrift vom Hintergrund noch hervorzuheben. Eine weitere Möglichkeit wäre es eine Box um die Schrift zu ziehen. Bei dieser Variante könnte man die Box auch transparent schalten, damit nicht zu viele Bildinformationen überdeckt werden. Nutzt man eine Freistellung, kann durchaus auch eine weiße Schriftfarbe gewählt werden, diese ist auch bei dunklen Flächen eine gute Alternative [Großer (2002)].

4.2.3 Schriftgrad

Der Schriftgrad wird auch als Schriftgröße bezeichnet und bietet eine weitere Möglichkeit, qualitative Unterscheidungen von Objekten vorzunehmen [Großer (2002)]. Um die Lesbarkeit zu gewährleisten, sollte eine Schriftgröße von 12 pt nicht unterschritten werden, da mobile Geräte, z. B. im Vergleich zu Laptops, ein recht kleines Display mit entsprechend schlechter Auflösung aufweisen [Brunner (2001)]. So sind bei einer Displayauflösung von 640 mal 800 Bildpunkten und einer Größe von ca. vier Zoll die Bildpunkte ca. 0,12 mm groß, was etwa 200 ppi (pixel per inch) entspricht [Brunner (2002)].

Auf Grund der geringen Displaygröe mobiler Geräte, wird sich die Schriftgröße sicherlich generell in dem Bereich von 12 pt ansiedeln. Um eine Differenzierung bezüglich der Entfernung von Gebäuden darzustellen, wäre es möglich, die Schriftgröße mit dem Abstand zum Nutzer zu variieren. So könnte ein nahegelegenes Objekt eine größere Schrift erhalten als ein Objekt, welches in großer Entfernung liegt.

Eine zu große Schriftsperrung sollte bei mobilen Anwendungen vermieden werden, da sie mehr Platz in Anspruch nimmt und somit wichtige Objekte verdecken kann. Eine Schriftsperrung ist an sich eher bei Beschriftungen von Flächen, wie z. B. Naturschutzgebieten oder Gebirgen, sinnvoll. Um eine gute Lesbarkeit zu gewährleisten, sollte allerdings ein gewisser Abstand zwischen den Buchstaben vorliegen. Es gibt spezielle Schriftfonts, welche direkt für das Internet und das Lesen am Bildschirm entwickelt wurden, sogenannte WebFonts, zu denen die Schrift Verdana zählt. Diese Schriften besitzen einen festen Buchstabenabstand und erhöhen dadurch die Lesbarkeit [Schulz (2004)].

In Abb. 4.7 wird verdeutlicht, welche Rolle die Schriftgröße sowie die Schriftart spielen. Eine Schriftgröße von 12 pt stellt in jedem Fall eine gute Lösung dar.

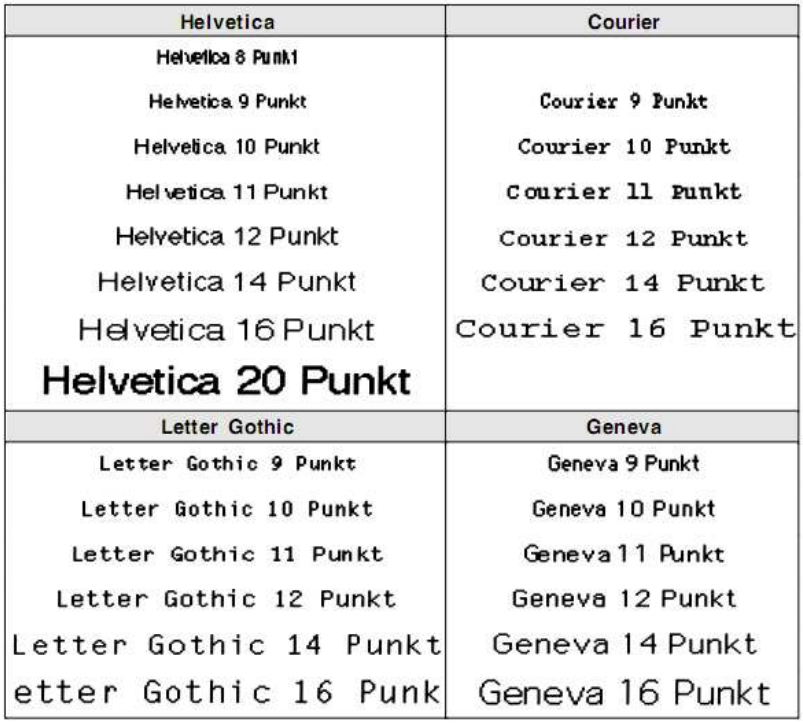


Abb. 4.7: Verschiedene Schriften mit unterschiedlichen Größen [Brunner (2001)]

5 Erstellung eines Campusführers

Die praktische Aufgabe dieser Diplomarbeit sieht die Erstellung einer Anwendung für Smartphones vor, welche auf dem Modell Hero von HTC getestet wird. Entstehen soll eine App, welche es ermöglicht, in einer AR Anwendung die Gebäudenamen auf dem Campus der Technischen Universität Dresden anzuzeigen. Die Anzeige von POIs lässt sich verhältnismäßig einfach realisieren. So können mittlerweile über Plattformen im Internet (z. B. Wikitude [Mobilizy (c)]) recht einfach Punkte eingetragen und verlinkt werden. Die besondere Herausforderung bei dieser Arbeit besteht darin, nur die Gebäude anzuzeigen, welche auch wirklich vom Standort des Nutzers aus gesehen werden können. Es ist also eine Analyse der Sichtbarkeit notwendig, um diese Anwendung realisieren zu können. Dieses Kapitel geht als erstes auf die Genauigkeit der Positionsbestimmung mittels GPS ein und betrachtet anschließend die einzelnen Arbeitsschritte (siehe Anhang A) und ihre Probleme. Zum Abschluss wird der Campusführer vorgestellt und bewertet.

5.1 Genauigkeit der Positionsbestimmung

Bevor es um die Arbeitsschritte bei der Erstellung des Campusführers gehen soll, muss man sich als erstes mit den Genauigkeiten auseinandersetzen, welche für diesen realistisch sind. Zunächst wurde die Genauigkeit der Positionsbestimmung des Smartphones überprüft. In dem zur Verfügung stehenden Smartphone HTC Hero kann eine Bestimmung der aktuellen Position mittels eines GPS-Empfängers, welcher sich im Handy befindet, eines drahtlosen lokalen Netzwerkes und des Mobilfunkstandards GSM erfolgen.
Die Bestimmung der Position des Smartphones erfolgte mit Hilfe der App „HTC Footsprints“ (Abb. 5.1), welche kostenlos zur Verfügung steht. Mit Hilfe des GPS-Empfängers kann die Lage des Standpunktes in geographischen Koordinaten angezeigt werden. Bei dieser Anwendung kann der Nutzer

ein Foto der aktuellen Position machen, z. B. einer Sehenswürdigkeit, dieses mit den zugehörigen GPS-Koordinaten verlinken und sich auf Google Maps den Standort anzeigen lassen.

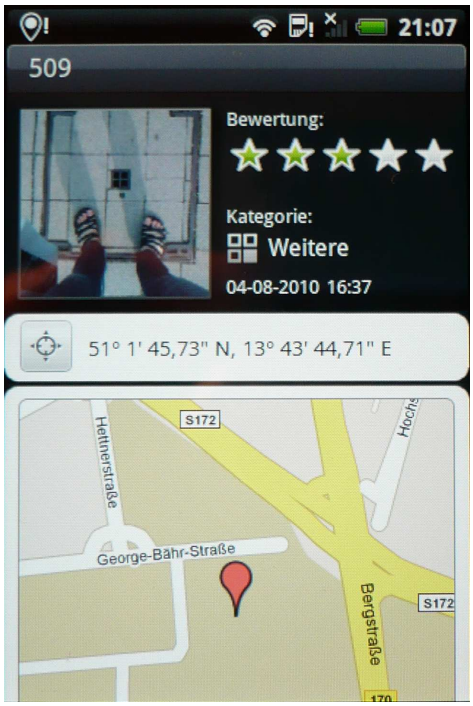


Abb. 5.1: Aufnahme der Koordinaten mit der Anwendung Footprints auf dem Smartphone

Die angezeigten Koordinaten sind geographisch, basierend auf dem World Geodetic System 1984 (WGS84) sowohl als Bezugssystem, als auch als Geodätisches Datum. Sollwerte wurden vom Geodätischen Institut der Technischen Universität Dresden zur Verfügung gestellt. Diese wurden tachymetrisch aufgenommen und sind deshalb wesentlich genauer als GPS-Koordinaten. Die Sollwerte haben eine Standardabweichung von 0,013 m. Sie liegen als Gauß-Krüger-Koordinaten mit dem Bezug European Terrestrial Reference System 1989 (ETRS89) vor. Um die Umrechnung der gemessenen geographischen Koordinaten zu vereinfachen, werden sie in UTM-Koordinaten umgewandelt (UTM: Universal Transverse Mercator). Dafür wurde das Programm EasyTrans verwendet, welches als eine kostenlose Testversion heruntergeladen werden kann [EasyTrans].

Wieso kann einfach eine Umrechnung der Koordinaten in UTM erfolgen, obwohl die Sollkoordinaten in der Gauß-Krüger-Abbildung vorliegen? Das WGS84 wird in regelmäßigen Abständen aufdatiert, damit es mit dem hochgenauen International Terrestrial Reference System (ITRS), welches auch Plattenbewegungen der Erde berücksichtigt, übereinstimmt. Dies erfolgte zum letzten Mal im Jahr

2009. Das ETRS89 entspricht den Messungen des ITRS aus dem Jahr 1989, auch WGS84 basiert auf diesen Werten. Seitdem hat sich die Lage Europas allerdings um ca. 50 cm verändert. Dies wurde bei den Sollkoordinaten, welche im ETRS89 vorliegen, nicht beachtet. Da man allerdings schon vor der Messung der Koordinaten durch den GPS-Empfänger des Smartphones davon ausgehen kann, dass die Genauigkeit ungefähr im 15 m Bereich liegt, sind diese 50 cm durchaus zu vernachlässigen. Um weitere grundlegende Informationen zum Thema Bezugssysteme und Geodätisches Datum zu erhalten, empfiehlt sich ein Blick in die Kartographischen Nachrichten Ausgabe 4 [Kreitlow u. a. (2010)] sowie in die Fachliteratur des Vermessungswesens [Torge (2003)].
Die Sollwerte sowie die gemessenen Werte, zum einen gegen Mittag, zum anderen am Nachmittag, finden sich in Tabelle 5.1.

Tab. 5.1: Unterschiede der gemessenen Koordinaten zu den Sollkoordinaten

Punktnr.	Sollwerte		GPS-Koordinaten			
			Mittags		Nachmittags/Abends	
	Rechtswert	Hochwert	Ostwert	Nordwert	Ostwert	Nordwert
500	410587,05	5653811,08	410587,53	5653825,02	410585,24	5653801,81
501	410687,00	5653809,31	410682,15	5653812,71	410661,42	5653789,82
505	410760,93	5653820,71	410760,19	5653823,26	410759,21	5653831,06
507	410825,23	5653828,72	410821,03	5653836,56	410795,57	5653865,03
509	410909,08	5653826,13	410834,33	5653843,45	410878,24	5653859,37
529	410834,49	5653712,60	410833,20	5653712,77	410847,89	5653759,12
524	410759,36	5653741,32	410766,39	5653767,65	410767,29	5653730,70
518	410629,69	5653738,19	410609,08	5653699,96	410640,12	5653734,68
516	410667,88	5653625,51	410638,46	5653703,67	410676,72	5653608,13
513	410842,11	5653626,68	410852,97	5653618,77	410843,50	5653655,31
532	410817,49	5653935,64	410814,37	5653930,33	410820,71	5653948,13

Mit Hilfe dieser Werte wurden Standardabweichungen für die mit dem Smartphone gemessenen Koordinaten berechnet. Dabei wurden zwei verschiedene Vorgehensweisen genutzt. Eine Möglichkeit, die Standardabweichung für die Messungen zu ermitteln, ist erst einen Mittelwert aus den Mittags- und Nachmittagsmessungen zu bilden und mit diesen die Genauigkeit zu berechnen. Auf diese Weise erhält man eine Abweichung von 18,23 m für den Hochwert und 18,89 m für den Rechtswert. Diese Berechnung kommt einer geodätischen Aufnahme nahe, bei denen Langzeitmessungen mit einem GPS-Empfänger durchgeführt werden. So stehen Vermessungsingenieure für eine gewisse Dauer an einem Ort und erhalten fortwährend Satellitendaten. Dies führt zu exakteren Werten durch Mittelung der berechneten Positionen.

Zum anderen wurden die Standardabweichungen einzeln für die Rechts- und Hochwerte sowie für die Messungen mittags und nachmittags berechnet. Der Hochwert hat am Nachmittag eine Standardabweichung von 25,62 m, der Rechtswert von 17,13 m. Die am Mittag gemessenen Werte haben eine Standardabweichung von 29,88 m des Hochwertes, sowie 26,64 m des Rechtswertes. Daraus ist ersichtlich, dass einerseits die Werte mittags schlechter sind als die nachmittags gemessenen Koordinaten. Andererseits sind die Rechtswerte ein wenig genauer als die Hochwerte.
Woran kann dies liegen? Durch die Bewegung der Satelliten auf ihren Orbits verändert sich die Satellitenkonstellation je nach Tageszeit, was zu unterschiedlichen Genauigkeiten führt. Außerdem ist es wichtig, an welchem Ort die Messung durchgeführt wird. In stark bebauten Bereichen, um welche es sich bei dem Gelände der Universität definitv handelt, ist der Empfang der Signale schwieriger als auf freien Flächen. So werden Signale durch die Gebäude abgelenkt und erreichen den Empfänger nicht direkt. Dieses Phänomen wird auch als Mehrwegeausbreitung bezeichnet [Wildt (2006)].
Die nach Tageszeit getrennte Berechnung der Standardabweichung kommt der realen Situation näher. So wird auch der Nutzer des Smartphones den GPS-Empfänger erst einschalten, bevor die Anwendung gestartet wird, da sonst die Akkulaufzeit sehr stark verringert wird. Der GPS-Empfänger hat dadurch zu wenig Zeit, die Position exakter zu bestimmen, was zu ungenaueren Ergebnissen führt.
Aus beiden Berechnungen kann festgestellt werden, dass mit dem genutzten Smartphone eine Positionbestimmung mit einer Genauigkeit von ungefähr 20 m erreicht werden kann.

5.2 Arbeitsschritte zur Erstellung der Sichtbarkeitsanalyse

Die Problemstellung der Sichtbarkeitsanalyse ist mit Hilfe eines Geoinformationssystems, wie in Kapitel 3 ersichtlich, relativ gut gelöst. Zur Lösung dieser Aufgabe wurde die Software ArcGIS 10 genutzt. Um eine Sichtbarkeitsanalyse in einem GIS-Programm durchzuführen, benötigt man ein Digitales Geländemodell oder, im Fall einer Sichtbareitsanalyse mit Gebäuden, ein Digitales Oberflächenmodell. Dieses wurde von der Firma Milan Geoservice GmbH [Milan] zur Verfügung gestellt. Aufgenommen wurde es mittels Airborne Laserscanning (ALS) [Wehr und Lohr (1999)] im Jahr 2005. Das DOM hat eine Auflösung von 4 Punkten pro m^2 . ArcGIS führt eine Sichtbarkeitsanalyse nur mit Rasterdaten durch, deshalb war es notwendig, ein gerastertes DOM zu nutzen.

Ein DGM ist nicht ausreichend für diese Aufgabe, da die Gebäude die wichtigste Rolle spielen. Während bei einem DGM nur das Gelände dargestellt wird, enthält ein DOM auch Bewuchs und Bebauung. Es wäre wünschenswert gewesen, wenn die Vegetation, vor allem Bäume, aus dem DOM herausgerechnet werden könnten, da Bäume oft die Sicht auf ein Gebäude versperren. Im Winter ist die Sicht jedoch besser, da die geringere Belaubung sie nicht so stark beeinträchtigt. Dies wurde bei der Lösung der Aufgabe nicht berücksichtigt, da sich das Herausrechnen der Vegetation als nicht trivial erwies (vgl. Abschnitt 5.6). Im Frühjahr und Sommer ist es außerdem wichtig, die Vegetation zu beachten, weil sie in vielen Fällen die Sicht auf Gebäude versperren. Abb. 5.2 zeigt das genutze DOM als 3D-Ansicht in ArcScene.

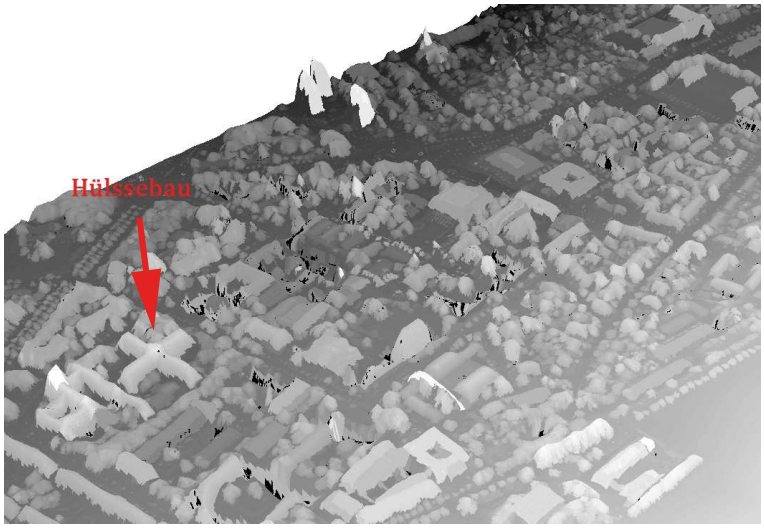


Abb. 5.2: DOM des Testgebietes

Neben einem Raster des Geländes benötigt man zur Durchführung der Analyse der Sichtbarkeit Standorte, welche in einem Shapefile mit dem Feature Type Point gespeichert sind. Das genutzte Shape beinhaltet ca. 1200 Standorte auf dem Campus der TU Dresden. Die Punkte liegen in einem Raster von 20 m. Dieser Abstand wurde gewählt, da zum einen der GPS-Empfänger im Smartphone keine höhere Genauigkeit aufweist, sowie die Datenmenge, welche aus den Berechnungen resultiert, in einem akzeptablen Rahmen bleibt. Die Rasterzellen wurden ebenfalls als Shapefile erzeugt und besitzen alle eine spezielle ID, welche bei weiteren Arbeitsschritten benötigt wird. Abb. 5.3 zeigt die Standorte der Beobachtungspunkte sowie die Rasterzellen, welche über dem Gelände des Campus liegen.

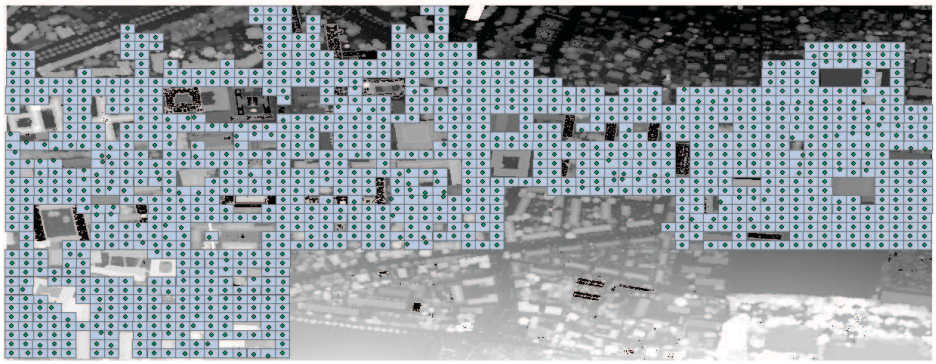


Abb. 5.3: Beobachtungsstandorte und 20 m-Raster über dem Campus

Die Funktion der Sichtbarkeitsanlyse (Viewshed) befindet sich in ArcGIS unter den Spatial Anlyst Tools. Da als Ergebnis wieder ein Raster entstehen soll, sind einige weitere Arbeitsschritte notwendig, bis hin zu einem Ergebnis, welches in die Geodatenbank PostGIS eingespielt werden kann. Das entstandene Raster weist für alle sichtbaren Bereiche einen Wert von 1 auf. Nichtsichtbare Gebiete erhalten den Wert 0. Es wird also eine binäre Analyse der Beobachtungspunkte durchgeführt (vgl. Abschnitt 3.2). Da in die Datenbank nur ein Shapefile eingespielt werden kann, muss eine Umwandlung des Rasters in ein Shapefile erfolgen. Um dieses Problem lösen zu können, nutzt man die Funktion Raster to Polygon, welche sich unter den Conversion Tools befindet. Da nur die sichtbaren Gebiete benötigt werden, sollen die Bereiche, welche nicht sichtbar sind, gelöscht werden. Dazu ist die RasterID behilflich, welche das Shapefile auch nach der Umwandlung vom Raster zum Polygon aufweist. Somit können alle Bereiche, welche den Wert 0 besitzen, gelöscht werden. Dies führt auch zu einer Reduzierung der Datenmenge.

Weiterhin soll jedes Shapefile eine ObserverID zugewiesen bekommen, mit der man die entsprechende Sichtbarkeitsanalyse später aufrufen kann. Diese orientiert sich an der Nummer der Rasterzelle, welche, wie oben beschrieben, eine ID aufweist, je nachdem, wo sie sich im Raster über dem Campus befindet. Diese ObserverID wird später in der Datenbank PostGIS für den Structured Query Language (SQL)-Request (vgl. Abschnitt 5.3) benötigt. Abb. 5.4 zeigt alle Sichtbarkeitsanalysen zusammengefasst sowie die Attributtabelle mit den ObserverIDs.

Die einzelnen Arbeitsschritte sind mit relativ wenig Zeitaufwand einzeln durchführbar. Da allerdings, wie erwähnt, um die 1200 Standorte vorliegen und für jeden einzelnen die verschiedenen Schritte durchgeführt werden müssen, wurde nach einer Möglichkeit gesucht, die jeweiligen Arbeitsvorgänge

automatisch durchzuführen. In der ArcToolbox lassen sich eigene Models aus verschiedenen Werkzeugen erstellen. So wurden alle Arbeitsschritte miteinander verknüpft und eine Schleife eingebaut, damit jeder Standort einzeln berechnet wurde. Das Model (Abb. 5.5) erzeugt am Ende für jeden Standort ein Shapefile mit den sichtbaren Bereichen. Diese können dann mit der Funktion Append in einem Shapefile abgespeichert werden (Abb. 5.4). Wichtig dabei ist, dass die zusammenzufügenden Dateien die gleichen Eigenschaften in der Attributtabelle aufweisen.

Dieses Shapefile, welches alle sichtbaren Bereiche beinhaltet, kann nun mittels der freien GIS-Software GRASS GIS in die Datenbank PostGIS eingespielt werden.

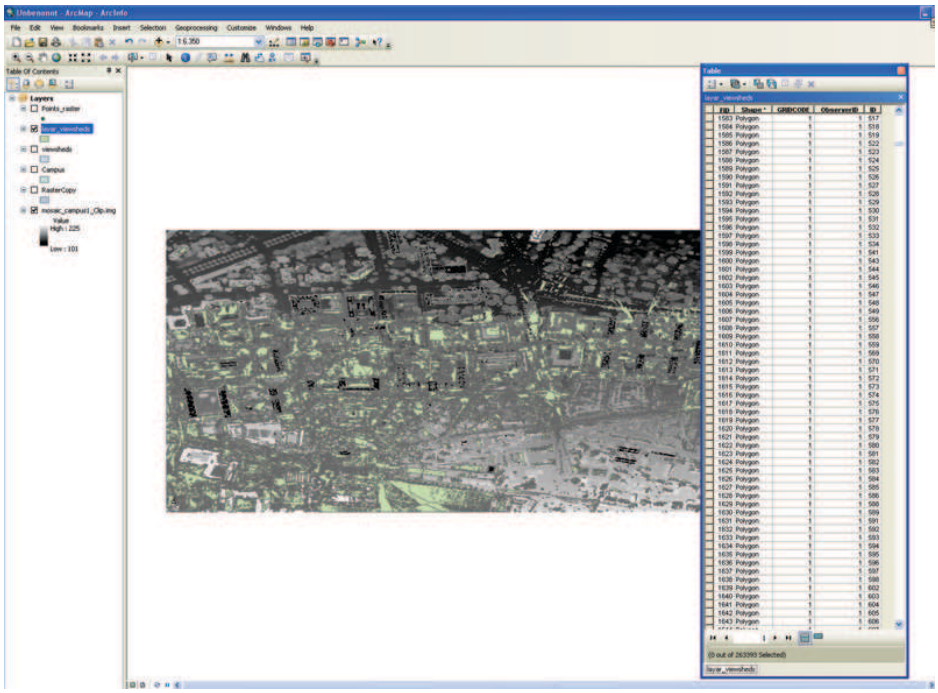


Abb. 5.4: Alle Sichtbarkeitsanalysen in einem Shapefile vereint

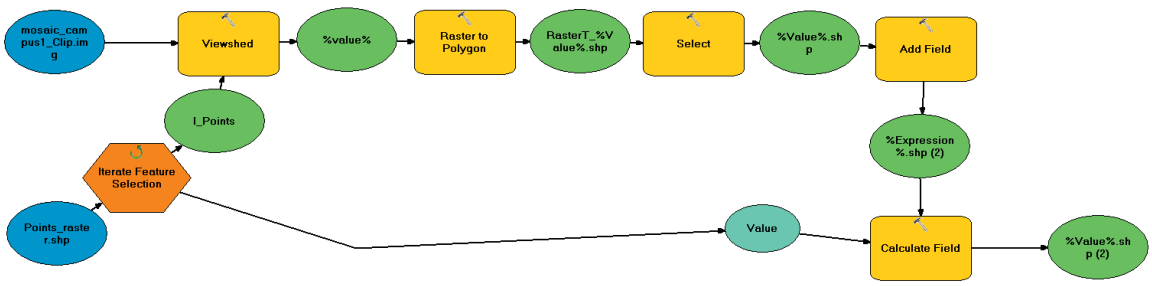


Abb. 5.5: In ArcToolbox erstelltes Modell

5.3 Arbeitsschritte in der Geodatenbank PostGIS

In die Datenbank müssen als erstes unterschiedliche Layer eingespielt werden, um danach verschiedene Abfragen durchführen zu können. So wird neben der Ebene, welche die Sichtbarkeitsanalysen beinhaltet, auch ein Layer mit den Grundrissen der Gebäude und das Shapefile mit den 20 m Rasterzellen benötigt. Die Gebäudegrundrisse wurden von der Plattform Open Street Map (OSM) [OSM] heruntergeladen.

Der GPS-Empfänger liefert die notwendige Position, an der sich der Nutzer befindet. Diese wird für die Abfrage der Datenbank benötigt. Gesucht werden die Namen der Gebäude, welche man von dieser Position aus sehen kann. Um diese herauszufinden, werden die Polygone der Sichtbarkeitsanalyse mit den Gebäudegrundrissen geschnitten. Dabei entstehen unter Umständen viele kleinere Polygone, da zum Teil nur Ecken von Gebäuden sichtbar sind oder ein Turm und nicht das zugehörige Haus. Aus den Polygonen, welche alle zu einem Gebäude gehören, also eine entsprechende ID für dieses vorweisen, wird der Schwerpunkt berechnet. Dieser Punkt des Gebäudes wird dann später auf dem Smartphone erscheinen.

Abb. 5.6 verdeutlicht, was die SQL-Abfrage bewirkt. Dargestellt ist das Ergebnis einer Sichtbarkeitsanalyse für einen Standort. Das Polygon schneidet den Gebäudegrundriss und daraus wird der Schwerpunkt gebildet.

Die SQL-Abfrage befindet sich im Java-Code, das heißt, die Datenbank PostGIS stellt die Daten bereit und erhält dann über das erstellte Java-Programm (vgl. Abschnitt 5.5) eine entsprechende Abfrage, bei der die genannten Arbeitsschritte durchgeführt werden.

5.4 Die Plattform Layar

Um die Anwendung erstellen zu können, wurde die Plattform Layar [Layar (b)] genutzt, welche eine Möglichkeit zur Erstellung von eigenen Layern für Smartphones aufweist. Diese AR Software war ursprünglich für Smartphones, welche über ein Android-Betriebssystem verfügen, gedacht. Mittlerweile ist es aber auch für iPhones, welche das iOS-Betriebssystem von Apple besitzen, verfügbar. Der Layar Reality Browser lässt sich kostenlos herunterladen. Neben schon vorhandenen Anwendungen im Bereich AR bietet sich auch die Möglichkeit, über diese Plattform eigene Layer

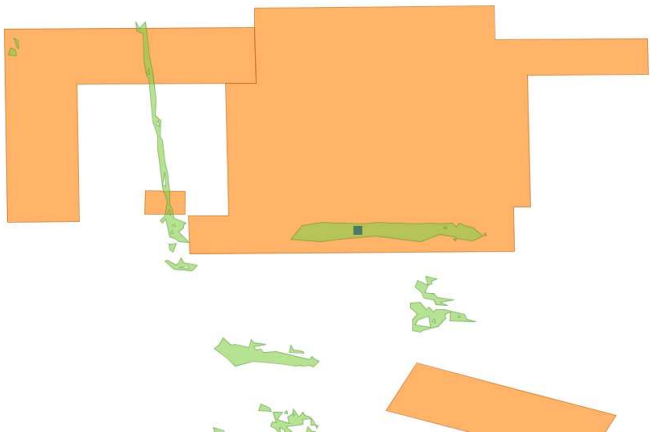


Abb. 5.6: Verschneidung der Sichtbarkeitspolygone (grün) mit Gebäuden (orange) sowie zugehöriger Schwerpunkt (blau)

zu erstellen und hochzuladen. Layar stellt also eine Programmierschnittstelle (application programming interface (API)) zur Verfügung, über welche man seine eigenen Anwendungen erzeugen kann. Das heißt, die Plattform Layar deckt zwei Bereiche ab, zum einen den Browser, um Apps zu starten und zu nutzen und zum anderen die Möglichkeit, über eine Schnittstelle eigene Anwendungen zu erzeugen.

Um einen eigenen Layer erstellen zu können, muss man sich zunächst als Entwickler bei Layar anmelden, also einen Account erstellen. Der Entwickler kann sich in einer Dokumentation über die Struktur der API informieren und Tutorials bearbeiten, sowie Beispiele für requests ansehen. Außerdem gibt es ein Diskussionsforum, in welchem man sich mit anderen Entwicklern austauschen und Hilfe suchen kann. Dieses sogenannte Wiki, eine Sammlung von Informationen, welche von Benutzern online geändert werden kann, stellt sehr viele Rahmenbedingungen und Hilfestellungen für die Entwicklung einer App bereit.

Ist man als Entwickler bei Layar angemeldet, kann man auf seine eigene Developer-Seite zugreifen, von der aus man eigene Layer erzeugen, verändern und testen kann. Weiterhin hat man Zugriff auf sämtliche wichtigen Informationen für das Erstellen einer eigenen App [Layar (b)].

Warum fiel die Entscheidung zugunsten Layars? Neben dieser Plattform gibt es eine ähnliche namens Wikitude, über welche man ebenfalls eigene Anwendungen erzeugen kann. Es wurde sich gegen diese entschieden, da zu Beginn der Diplomarbeit Wikitude noch nicht ausreichend entwickelt war, dass eine solche Anwendung über diese Plattform erzeugt werden konnte [Mobilizy (a)]. Mittlerweile ha-

ben sich beide Plattformen in großem Maße weiterentwickelt. Darauf soll im Ausblick dieser Arbeit (vgl. Kapitel 6) noch einmal genauer eingegangen werden.

Hat man einen Entwickler-Account angelegt, kann man seinen eigenen Layer erstellen und diesen editieren. So sollten z. B. einige Informationen über den Zweck des Layers angegeben werden. Im Fall der vorliegenden Diplomarbeit handelt es sich um einen Layer, welcher die Gebäudenamen des Campus anzeigt, allerdings nur solche, welche auch wirklich von einer bestimmten Position sichtbar sind. Als erstes muss ein einheitlicher Quellenanzeiger (Uniform Resource Locator (URL)) angegeben werden, welcher die Punkte abruft, die über das Java-Programm dem Server zur Verfügung gestellt werden. Weiterhin können verschiedene Optionen gewählt werden, beispielsweise, ob der Layer Sounds abspielen soll, 3D-Objekte beinhaltet oder Animationen anzeigen soll. Außerdem kann das Layout, welches auf dem Display des Smartphones erscheint, verändert werden, und entschieden werden, welche Icons die POIs besitzen sollen. In Abb. 5.7 ist dargestellt, was über die Benutzeroberfläche der Plattform Layar verändert werden kann.

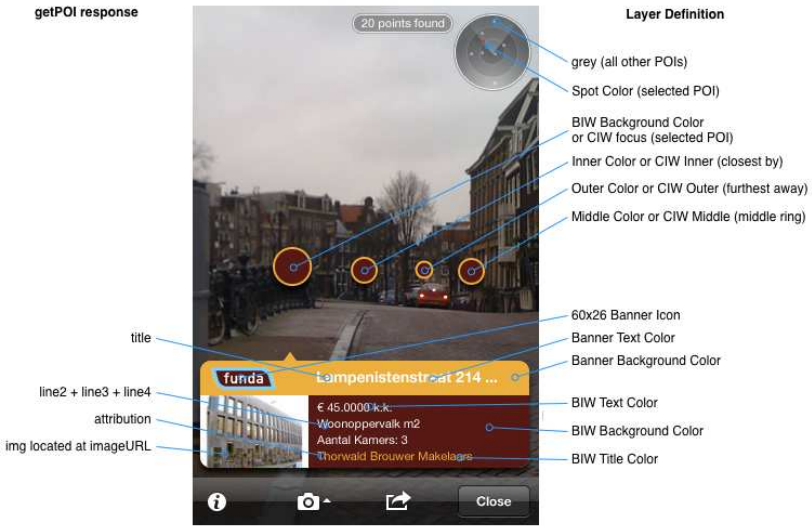


Abb. 5.7: Über den Entwickleraccount individuell veränderbares Layout [Layar (c)]

Um die Icons für die Anzeige der Punkte auf dem Display zu verändern, sollten bestimmte Kriterien erfüllt sein. So müssen sie eine bestimmte Größe haben, je nachdem, für welche Position sie gedacht sind. Fokussierte Punkte sollen beispielsweise eine Icongröße von 45x45 Pixel aufweisen, wohingegen Punkte, die sehr weit entfernt sind, lediglich 20x20 Pixel benötigen. Außerdem muss das selbst entworfene Icon einen Rand aus einem weißen Pixel erhalten. Danach folgt ein schwar-

zer Pixelrahmen mit 40 % Opazität sowie vier Pixel mit einer Opazität von 40 % bis 0 %. Um diese Vorschriften umsetzen zu können, wurde die kostenlose Software IcoFX [Kovrig] zur Erstellung von Icons benutzt. Schriften direkt in das Anzeigefeld, also anstelle eines Icons, einzubinden, ist über die Layarplattform nicht möglich. Am unteren Bildschirmrand befindet sich, wie in Abb. 5.7 ersichtlich, ein Anzeigefeld, in welchem der Name des POIs auftritt. Um dennoch eine Beschriftung des Punktes zu erreichen, besteht die Möglichkeit, die Kürzel der Universitätsgebäude als Icons zu erstellen. Testweise erfolgte dies für ein ausgewähltes Gebäude, so wurde für den Hülssebau das Kürzel HÜL verwendet (vgl. Abbschnitt 5.7).

Neben den Einstellungen für das Layout lassen sich auch verschiedene Filter [Wang (2011)] einfügen. Verwendet werden kann z. B. eine Suchbox, in welche man eingeben kann, wonach man Ausschau hält. Über einen Range-Slider lässt sich der Radius verschieben, in welchem die POIs angezeigt werden sollen.

Darüber hinaus lassen sich noch spezifische Kriterien durch die Filter einbinden. Als Beispiel dient die Wohnungssuche. Vorstellbar ist eine Anwendung, welche die zu vermietenden Wohnungen anzeigt. Mit der Suchbox kann man beispielsweise nach einer ganz bestimmten Straße suchen. Weiterhin kann ein Filter eingebaut werden, eine sogenannte Checkbox-List, mit welchem man bestimmte Kriterien auswählen kann. Z. B., ob man nach einem Haus, nach einem WG-Zimmer oder einer 3-Zimmer-Wohnung sucht. Mit einer Radiobutton-List lässt sich in diesem Beispiel auswählen, ob die Wohnung zur Miete oder zum Kauf gedacht sein soll. Mit einem Custom-Slider kann man anschließend einen Bereich auswählen, welcher finanzielle Rahmen minimal und maximal für eine Wohnung angedacht ist. Abb. 5.8 verdeutlicht die möglichen Filter. Ein Einsatz eines solchen Filters für die Anwendung des Campusführers wurde als nicht sinnvoll erachtet. Möglich wäre der Einsatz der Funktion Textbox, um nach einem bestimmten Gebäude zu suchen. Allerdings werden nur die in diesem Moment sichtbaren Gebäude angezeigt, wodurch die Anwendung nur deren Namen anzeigen würde, aber keine Gebäudenamen außerhalb des Sichtbereiches. Einen Radius zu bestimmten, in welchem sich das Gebäude befinden soll, ist ebenfalls nicht sinnvoll, da, wie bereits erwähnt, nur sichtbare Objekte angezeigt werden, welche sich auf dem TU Gelände nicht in allzu großer Entfernung finden lassen. Bei solchen geringen Strecken, wie sie in der entwickelten Anwendung vorkommen, sollte man ebenfalls überlegen, ob dies auf Grund der Ungenauigkeit der Positionsbestimmung überhaupt einen Nutzen vorweist.

Eine große Erleichterung bei der Entwicklung der Anwendung ist, dass über Layar bereits die Ausrichtung des Handys bestimmt wird. Diese Berechnungen müssen also nicht vom Entwickler bedacht werden. Auch der Öffnungswinkel der Kamera sowie Kippungen und Schwenkungen werden durch Layar bereits berücksichtigt und müssen nicht implementiert werden.

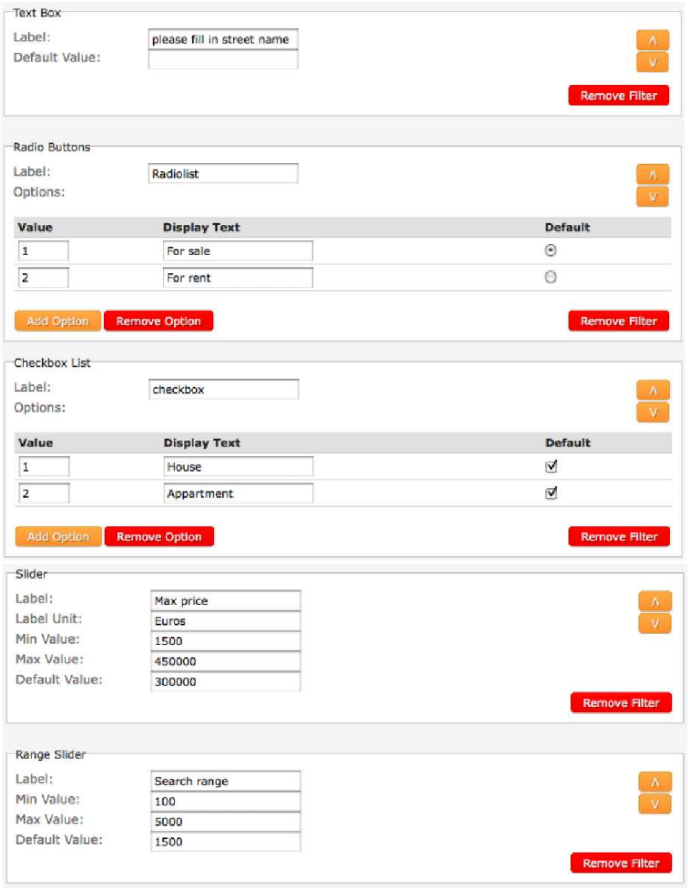


Abb. 5.8: Mögliche Filter der Plattform Layar am Beispiel eines Apps für die Wohnungssuche [Wang (2011)]

5.5 Die Programmierung mit Java

Dank der umfassenden Dokumentation im Layar-Wiki konnte die Umsetzung mit Hilfe der Programmiersprache Java in der Entwicklungsumgebung Eclipse erfolgen, in welcher ein GetPointsOfInterest-Request, ein Query-Builder, ein ifkTools2-0.0.1-SNAPSHOT Java Archiv sowie eine Extensible Markup Language (XML)-Datei erstellt werden musste.

Die Bibliothek ifkTools2 wird bei der Erstellung des Codes benötigt. Diese kann, im Gegensatz zu allen anderen genutzten Bibliotheken nicht aus dem Internet heruntergeladen werden. Sie befindet sich deshalb, wie alle anderen Dateien auf der Daten-CD, welche dieser Arbeit beigelegt ist.
Die XML-Datei wird für den Apache Maven Builder benötigt. Apache Maven ist ein sogenanntes Build-Management-Tool, also ein Werkzeug für den Erstellungsprozess von Programmen. Es basiert auf der Programmiersprache Java. Mit ihm können Java-Programme standardisiert erstellt, verwaltet und kompiliert werden. Die Konfigurationsdatei heißt pom.xlm, wobei pom für Project Object Model steht. Diese Datei enthält alle Informationen zum Softwareprojekt und folgt außerdem einem standardisierten Format, welches beim Ausführen von Maven geprüft wird. Beinhaltet die Datei alle Angaben und sind diese syntaktisch korrekt, wird die Arbeit fortgesetzt. Mit Hilfe von Apache Maven wird auch auf die URL des Servers zugegriffen, auf welche die POIs gespielt werden und worauf Layar zugreift, um die Punkte auf dem Display des Smartphones visualisieren zu können.
Der GetPointsOfInterest-Request unterliegt sehr vielen Vorschriften seitens Layar. Man kann in der Dokumentation alle wichtigen Hinweise zur Erstellung einer solchen Abfrage nachschlagen. Um den Code zu erstellen, mussten einige wichtige Standards und Vorschriften erfüllt sein, auf die im Folgenden ein kurzer theoretischer Blick geworfen werden soll.
Begonnen wird mit der JavaAPI for RESTful Web Services (JAX-RS). Sie stellt eine Spezifikation einer API von Java dar, welche die Nutzung der Representational State Transfer (REST) Architektur ermöglicht. Was ist nun aber diese REST Architektur? Sie ist ein Modell, welches beschreibt, wie das Web funktionieren sollte und dient der Realisierung von Web Services. Es handelt sich hierbei weder um einen Standard noch um ein Produkt, sondern um eine Richtlinie, wie Web Standards angewendet werden sollten. Mit Hilfe dieses Konzepts ist eine Ressource über einen Web Server verfügbar und wird über ein einheitlichen Bezeichner für Ressourcen (Uniform Resource Identifier (URI)) identifiziert [Bayer (2002)].
Des Weiteren wurde das Open Source Projekt Jersey genutzt, welches eine Referenzimplemetierung für JAX-RS darstellt. Eine Referenzimplementierung stellt einen Anhaltspunkt bzw. eine Empfehlung für eine Implementierung eines Standards dar. Außerdem stellt das Jersey Projekt eine API zur Verfügung, damit Entwickler Erweiterungen erstellen können. Nähere Informationen dazu finden sich auf der Homepage des Projektes [Jersey] sowie im Wiki [Oracle-Corporation].
Weiterhin wurde die JavaScript Object Notation (JSON) verwendet, welches ein Format zum Austausch von Daten zwischen Anwendungen darstellt. Es liegt in einer sowohl für den Menschen, als

auch für den Computer lesbaren Textform vor. JSON ist von der Programmiersprache unabhängig. Sie stellt ein gültiges JavaScript dar und kann deshalb sofort ausgeführt und in ein JavaScript-Objekt überführt werden. Somit kann ein normaler Attributzugriff erfolgen, um die einzelnen Eigenschaften aufrufen zu können. Natürlich müssen auch hierbei bestimmte Formatdefinitionen beachten werden. So werden z. B. Objekte in geschweifte Klammern gesetzt und können eine ungeordnete Liste von Eigenschaften enthalten, die durch Kommas getrennt werden. Ein Array wird in eckige Klammern gesetzt und kann eine geordnete Liste von Werten enthalten, welche ebenfalls durch Kommas voneinander getrennt werden [JSON].

Tritt bei Ausführung des Programmes ein Fehler auf, kann nach der Probe über eine Test-URL (Abb. 5.9) die erhaltene Ausgabe in dem JSONLint Validator (http://www.jsonlint.com) überprüft werden. Dieser testet, ob die Anforderungen erfüllt werden und gibt entweder eine Erfolgs- oder eine Fehlermeldung aus (Abb. 5.10). Die Fehlermeldung hilft in vielen Fällen mehr als jene, welche über die Layar Testseite aufgeführt wird. So können Fehler einfacher behoben werden.


```
{
"layer": "tudresdencampusguide2",
"errorString": "ok",
"morePages": false,
"errorCode": 0,
"nextPageKey": null,
"hotspots": [
  {"distance": 1000, "attribution": "This is a test layer POI provider", "title": "Fritz-Foerster-Bau (FOE)", "lat":
51027880, "lon": 13727744, "imageURL": null, "line4": "RADIOLIST-None,CustSlider-None", "line3": "SEARCHBOX -
asdfdgxdg", "line2": "DevlId - 896Settings: range=1000", "actions": [], "type": 0, "id": "test_2"},

{"distance": 1000, "attribution": "This is a test layer POI provider", "title": "Binder-Bau (BIN)", "lat": 51027515,
"lon": 13726862, "imageURL": null, "line4": "RADIOLIST-None,CustSlider-None", "line3": "SEARCHBOX - asdfdgxdg",
"line2": "DevlId - 896Settings: range=1000", "actions": [], "type": 0, "id": "test_2"},

{"distance": 1000, "attribution": "This is a test layer POI provider", "title": "Technische Leitzentrale (TLZ)", "lat":
51028158, "lon": 13727011, "imageURL": null, "line4": "RADIOLIST-None,CustSlider-None", "line3": "SEARCHBOX -
asdfdgxdg", "line2": "DevlId - 896Settings: range=1000", "actions": [], "type": 0, "id": "test_2"},

{"distance": 1000, "attribution": "This is a test layer POI provider", "title": "Müller-Bau (MÜL)", "lat": 51028213,
"lon": 13727733, "imageURL": null, "line4": "RADIOLIST-None,CustSlider-None", "line3": "SEARCHBOX - asdfdgxdg",
"line2": "DevlId - 896Settings: range=1000", "actions": [], "type": 0, "id": "test_2"},

{"distance": 1000, "attribution": "This is a test layer POI provider", "title": "Walther-Pauer-Bau (PAU)", "lat":
51028484, "lon": 13727033, "imageURL": null, "line4": "RADIOLIST-None,CustSlider-None", "line3": "SEARCHBOX -
asdfdgxdg", "line2": "DevlId - 896Settings: range=1000", "actions": [], "type": 0, "id": "test_2"}
]
}
```


Abb. 5.9: Ausgabe des Testrequests

Der GetPointsOfInterest-Request wurde mit Hilfe all dieser Standards und APIs entwickelt. In ihm wird auch der QueryBuilder aufgerufen, welcher die SQL-Abfrage beinhaltet. Diese wurde nach den in Kapitel 5.3 genannten Arbeitsabläufen erstellt. Der gesamte Code für diese Anwendung ist im Anhang (B) dieser Arbeit zu finden.

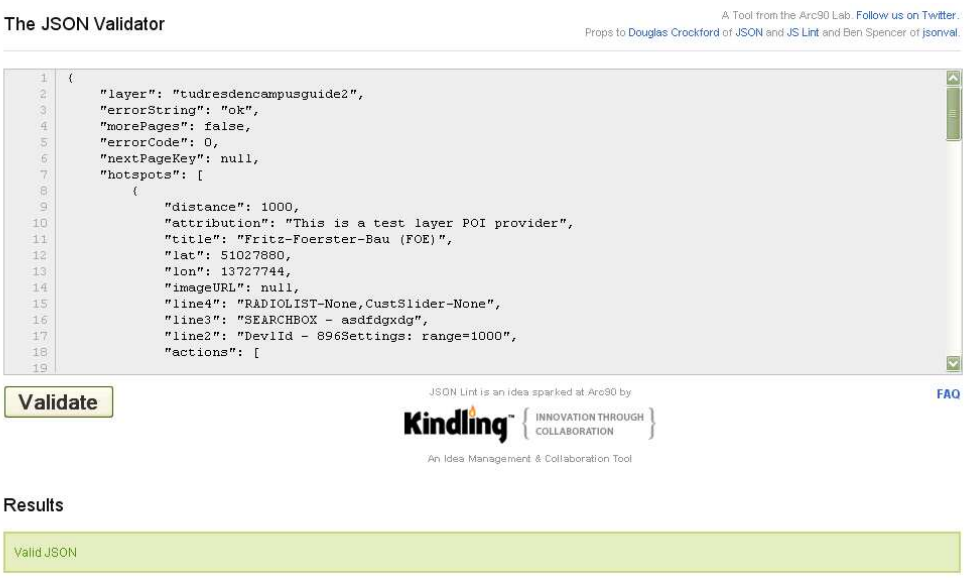


Abb. 5.10: Validierung des Testrequests mit Erfolgsmeldung

5.6 Probleme bei der Erstellung des Campusführers

Die Erstellung des Campusführers lief nicht ohne Probleme ab. Sie traten in jeder Phase der Entwicklung auf und konnten nur zum Teil behoben werden. Eine erste Hürde musste mit der Beschaffung der Daten genommen werden. Geprüft wurde zunächst eine Nutzung der Daten aus dem Projekt OpenStreetMap 3D [Universität Heidelberg]. Die Plattform OSM stellt prinzipiell ihre Daten zur freien Verfügung. Unter bestimmten Auflagen können diese vom Nutzer heruntergeladen und genutzt werden. Die 3D-Modelle dieses Projektes können allerdings nicht heruntergeladen werden, was eine Nutzung der Daten für die Erstellung des Campusführers ausschloss. Somit musste über Alternativen nachgedacht werden. Die Technische Universität Dresden hat ein CAD-Modell der Gebäude des Campus erstellt. Somit wäre diese Möglichkeit eine weitere Option gewesen, genau wie das 3D-Modell des städtischen Vermessungsamtes in Dresden. Doch beide Modelle konnten nicht verwendet werden, da sowohl ArcGIS als auch GRASS GIS für die Sichtbarkeitsanalyse Raster benötigen und die beiden Modelle als Vektordaten vorlagen. Eine Umwandlung der Vektordaten in Rasterdaten ist zwar in den GIS-Anwenderprogrammen gelöst, allerdings für den dreidimensionalen Bereich nicht ausreichend. Die gewonnen Daten waren nicht zufriedenstellend und konnten deshalb nicht genutzt werden. Zur Lösung des Problems musste demzufolge ein gerastertes DOM erzeugt bzw. beschafft werden. Letztendlich stellte die Firma Milan Geoservice GmbH ein solches Modell zur Verfügung.

Die Rasterdaten lagen leider nicht georeferenziert vor, sodass dieses durch die Software ERDAS Imagine bzw. ArcGIS erfolgen musste. Beide Programme lieferten ein ähnliches Ergebnis nach der Georeferenzierung, sodass beide geeignet sind, um diese Aufgabe zu lösen.
Wie bereits erwähnt, beinhaltet das DOM neben der Bebauung auch Vegetation, was unter Umständen das Ergebnis der Sichtbarkeitsanalyse verfälscht. So ist die Vegetation sehr stark von der Jahreszeit abhängig. Verdecken Bäume und Büsche im Frühjahr und Sommer zum Teil sehr große Teile von Gebäuden, ist die Sicht im Winter wesentlich freier. Außerdem kann unter Umständen ein Gebäude hinter einem Baum auch gesehen werden, wenn die Belaubung nicht ganz so stark ist. Es wurde dennoch versucht, ein Raster zu schaffen, welches nur das Gelände mit Gebäuden beinhaltet. Dabei wurden die Grundrisse der Gebäude genutzt und mit dem DOM verschnitten, sodass die Rasterdaten der Gebäude übrig blieben. Angedacht war eine Verrechnung des DGMs mit den Rasterdaten der Gebäude. Versucht wurde ebenfalls, ein Programm mit Hilfe von ArcObjects für ArcGIS zu erstellen, welches die Höhen der Dächer mit dem DGM verbindet und somit ein Modell erstellt, welches nur Gebäude beinhaltet. Beide Versuche konnten allerdings weder mit der Software ArcGIS noch mit FME realisiert werden. Die Optionen für dreidimensionale Berechnungen scheinen noch nicht weitreichend genug implementiert zu sein. Auf Grund dieser Schwierigkeiten wurde am Ende doch das DOM genutzt, welches Daten über Vegetation enthält. Bei einer Weiterführung dieses Projektes bzw. bei ähnlichen Projekten sollte versucht werden, dieses Problem zu lösen, im einfachsten Fall durch die Nutzung eines Rasters, welches keine Vegetation beinhaltet.
Im weiteren Ablauf ging es an die Bewältigung der Arbeitsschritte in ArcGIS. Diese sollten möglichst automatisiert ablaufen und in einem zeitlich vertretbaren Rahmen bleiben. Das heißt, eine Laufzeit von wenigen Stunden sollte möglich gemacht werden. Mit dem Modelbuilder der Software ArcGIS lies sich eine Verknüpfung der einzelnen Werkzeuge, welche zur Erfüllung der Aufgabe genutzt wurden, relativ leicht realisieren. Als erstes wurde eine Umsetzung mit ArcGIS 9.3 angestrebt, da diese als Studentenlizenz auch von zu Hause aus genutzt werden konnte. Dabei traten einige Probleme auf, so konnten z. B. Schleifen nicht einfach eingefügt werden. Um die Berechnung für jeden Standpunkt des Shapefiles einzeln durchführen zu können, war allerdings eine Schleife notwendig. Außerdem konnten gewisse Werkzeuge in der Studentenlizenz nicht eingesetzt werden, da diese gesperrt waren. Deshalb wurde auf die neue ArcGIS-Lizenz, welche im PC-Pool der Geowissenschaften der TU Dresden zur Verfügung stand, ArcGIS 10 zurückgegriffen. Bei dieser Version wurden die Funktionen, um ein eigenes Model für die Toolbox zu erstellen, erweitert und eine

Schleife konnte mühelos eingebaut werden. Das Erstellen des Models brachte nur wenige Probleme mit sich. Es musste die eine oder andere Funktion, welche zunächst vorgesehen war, ausgetauscht und durch eine bessere ersetzt werden, um den Ablauf zu optimieren. Im Endeffekt wurde aber eine Funktionskette erstellt, welche alle Aufgabe bewältigen konnte. Es konnten die notwendigen Shapefiles eingelesen und das Model gestartet werden. Hier zeigte sich schnell, dass die Bearbeitungszeit einige Stunden in Anspruch nehmen würde. Das Shapefile beinhaltete, wie bereits angedeutet, ca. 1200 Standorte. Diese mussten einzeln berechnet und abgespeichert werden. Nach ca. 1100 Punkten brach das Model die Berechnungen ohne Fehlermeldung ab. Abb. 5.11 zeigt das berechnete Gebiet. Auch bei weiteren Versuchen konnte nicht die komplette Anzahl an Punkten berechnet werden. Ein Einsetzen des Prozesses an der Stelle des Abbruchs war nicht möglich, da dadurch die ObserverIDs falsch vergeben wurden und eine Fehlinterpretation bei der SQL-Abfrage die Folge wäre bzw. bei der Speicherung der Daten einfach Informationen überspeichert werden würden und somit verloren gingen. Dieses Problem ist nach Ansicht der Diplomandin durch einen leistungsstärkeren Rechner bzw. durch ein zu erstellendes Programm lösbar. Jenes könnte mit Hilfe von ArcObjects diese Arbeitsschritte durchführen und ein individuelles Einsetzen an einer Stelle ermöglichen, an der auch die ObserverIDs geändert werden können und somit auch die Namen der Daten. Dies ist nötig, damit später die korrekte Zuordnung der einzelnen Sichtbarkeitsanalysen möglich ist.

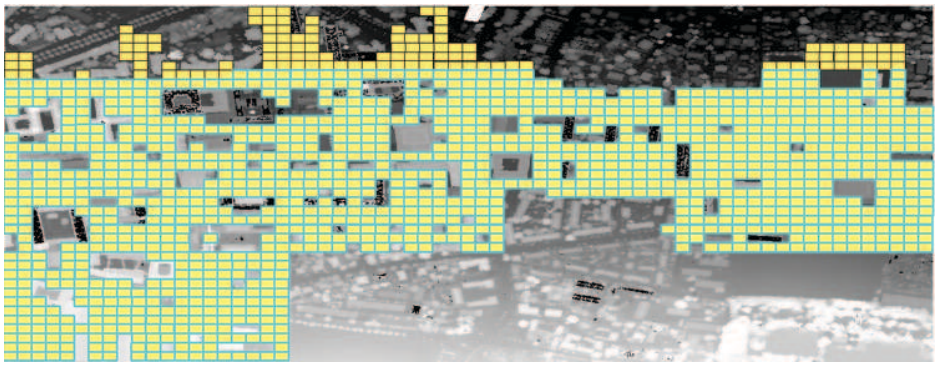


Abb. 5.11: Übersicht der bearbeiteten (blau umrandet) und unbearbeiteten Rasterzellen

Die Arbeit mit der Datenbank lief relativ problemlos ab. Auch hier musste die eine oder andere Methode getestet werden, bis eine endgültige Lösung gefunden wurde. Die Einspielung der Daten mit Hilfe von GRASS GIS verlief ebenfalls ohne Probleme. Dank der Schnittstelle zwischen den beiden Programmen konnten die Shapefiles aufgespielt und in der Datenbank genutzt werden.

Die Erstellung des Programmes mit Hilfe von Eclipse wies wesentlich mehr Probleme auf. Da von zwei verschiedenen Rechnern aus gearbeitet wurde, führte dies zu einigen Schwierigkeiten. Die Grundstruktur des Programmes wurde von Dipl.-Ing. (FH) Stefan Hahmann erstellt. Änderungen des Codes sowie die zusätzliche Einbindung von Funktionen konnten auf dem Laptop der Diplomandin vorgenommen werden. Es musste also die Verbindung zum Server des Institutes geschaffen und viele Einstellungen übernommen werden, welche nicht immer sofort funktionierten. Somit gab es hin und wieder Verzögerungen durch auftretende Fehler im Code sowie durch Ausfälle des Servers. Möchte ein Dritter Veränderungen am Code vornehmen, wäre es notwendig, die Entwicklungsumgebung Eclipse zu nutzen und die zahlreichen Einstellung innerhalb dieser vorzunehmen, um die Anwendung weiter auszubauen.

Die Veränderungen auf der Plattform Layar konnten ohne Schwierigkeiten vorgenommen werden. Die Erstellung der Icons war aufwendig, da viele Vorschriften eingehalten werden mussten. Waren diese einmal geschaffen, war es, dank der guten Dokumentation seitens Layar, kein Problem, diese für die Anwendung einzubinden. Wünschenswert wäre eine höhere Individualität bzgl. des Layouts. Der Entwickler ist in vielen Fällen an die Vorgabens Layars gebunden und kann nur in wenigen Fällen Veränderungen vornehmen. So wäre ein eigenes Layout der Anwendung sinnvoll, z. B. im Coorporate Design der TU Dresden. Eine große Vereinfachung stellt, wie bereits in Abschnitt 5.4 erwähnt wurde, die Vorberechnung der Schwenkungen, Kippungen und Ausrichtung sowie der Öffnungswinkel der Kamera seitens Layar, dar. Um diese Berechnungen, welche sehr aufwendig wären, muss sich der Entwickler keine Gedanken machen. Dies stellt einen hohen Grad an Komfort dar.

Zusammenfassend kann also festgestellt werden, dass fast alle Arbeitsabläufe Probleme mit sich brachten. Diese konnten allerdings in den meisten Fällen, wenn auch mit kleineren und größeren Kompromissen, behoben werden. Auf Verbesserungen innerhalb der Arbeitsabläufe und der Anwendung an sich soll später noch eingegangen werden (vgl. Abschnitt 5.8).

5.7 Der Campusführer

Die Layar-Developer Seite bietet neben den verschiedenen Editiermöglichkeiten auch eine Testseite für den erstellten Layer an. Auf dieser kann auf einer Karte getestet werden, ob die Punkte richtig angezeigt werden. Mit Hilfe einer kleinen Figur kann man den Standpunkt festlegen, von welchem aus die Anwendung gestartet werden soll. Z. B. stellt man die Figur vor die Alte Mensa, welche sich auf der Mommsenstraße befindet (Abb. 5.12) und erhält nach betätigen des „Load POIs“-Buttons die von dieser Position aus sichtbaren Gebäude. Auf der Testseite kann also die Richtigkeit der Anwendung überprüft werden. In einem Feld unter der Karte werden die Koordinaten der Position ausgegeben. Sollte ein Fehler beim Laden des Layers oder der POIs auftreten, wird dieser ebenfalls unterhalb der Karte mit einer entsprechenden Meldung angezeigt. Die Namen der Gebäude werden unterhalb der Karte am rechten Rand angezeigt (Abb. 5.13).

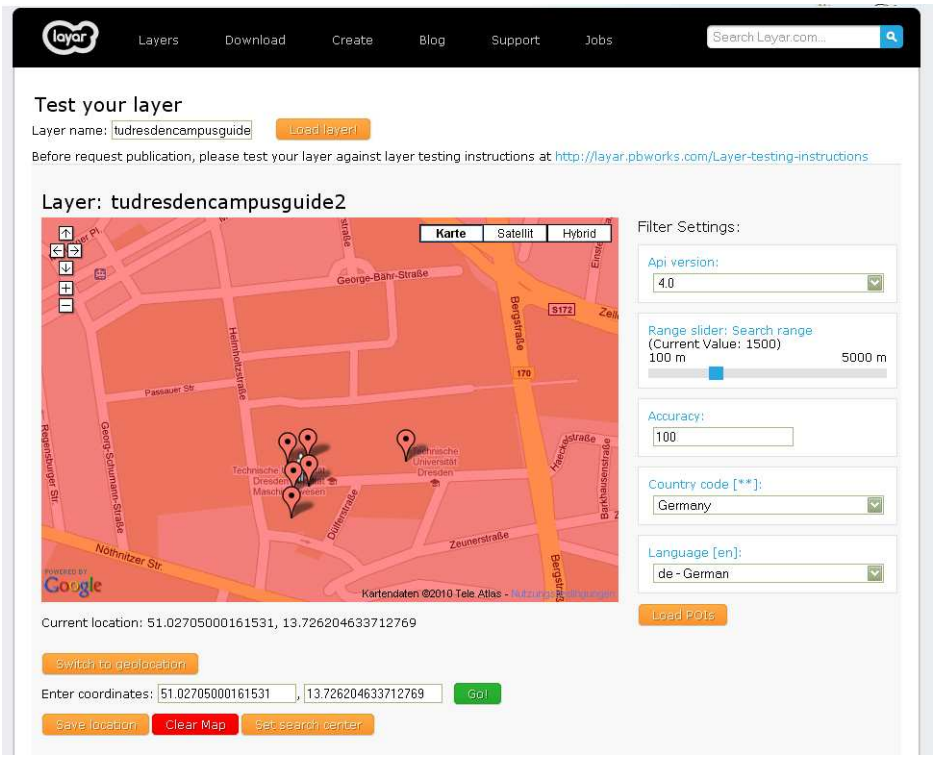


Abb. 5.12: Test des Campusführer-Layers mit Standpunkt vor der Alten Mensa

Aufwendig an dieser Testseite ist, dass die Figur beim Start immer in Amsterdam steht und von dort erst an die richtige Stelle verschoben werden muss. Es wäre wünschenswert, wenn dies in Zukunft von Seiten Layars komfortabler gestaltet werden würde.

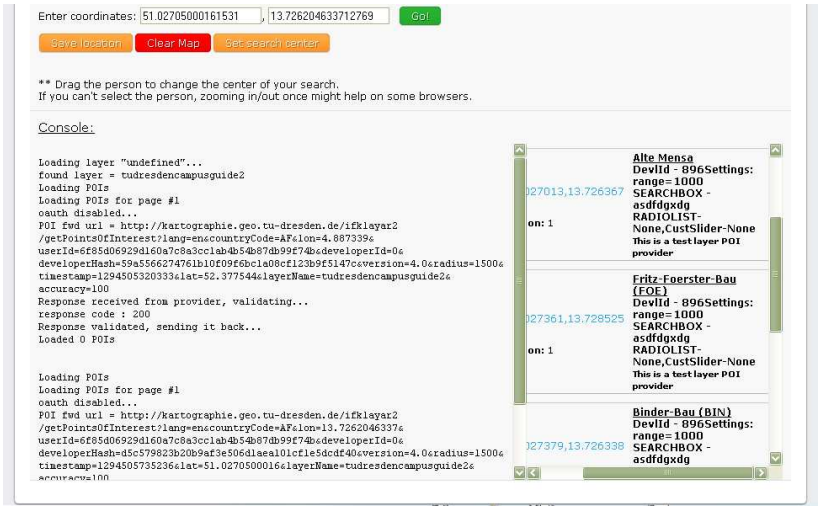


Abb. 5.13: Meldung bei Ausführung des Layers sowie Namen der Gebäude (rechts)

Testet man die Anwendung auf dieser Seite, stellt man fest, dass sie in den meisten Fällen die richtigen Ergebnisse anzeigt. Allerdings gibt es auch Positionen, von denen man als Ortskundiger weiß, dass man mehr Gebäude sehen müsste als angezeigt werden. Ein gutes Beispiel stellt eine Postion auf der Bergstraße dar. Von dieser aus sollten, neben der Neuen Mensa, auch das Hörsaalzentrum, der Chemiebau und der Gerberbau sichtbar sein. Angezeigt wird allerdings nur die Mensa (Abb. 5.14). Woran könnte dies liegen? Zu vermuten ist, dass sich etwas vor den Gebäuden befindet, was die Sicht auf diese versperrt. Es liegt nahe, die Bäume dafür in Betracht zu ziehen. Ist man ortskundig, kann bestätigt werden, dass sich sowohl vor dem Hörsaalzentrum als auch vor dem Chemiebau Bäume befinden, welche die Sicht versperren könnten. Vor dem Gerberbau wächst allerdings kein Baum. Dieser könnte wiederum von der Brücke oder einem Pfeiler verdeckt werden. Sieht man sich die Position der Figur genauer an, stellt man fest, dass sie höchstwahrscheinlich tatsächlich direkt neben der Brücke steht und somit keine Sicht auf den Gerberbau hat.

Ein anderes Beispiel zeigt genau das Gegenteil des vorhergehenden. In jenem Fall steht die Figur vor dem Trefftzbau, welcher jegliche Sicht in Richtung Osten versperrt. Dennoch wird die Bibliothek und das Biologiegebäude als sichtbar angezeigt (Abb. 5.15). Dies ist sehr unwahrscheinlich, aber auch hierfür gibt es eine Erklärung. Da die Vegetation im Raster vorhanden ist, kann es passieren, dass ein Standort, welcher vorher festgelegt wurde, auf dieser liegt und somit höher als das eigentliche Gelände ist. Der Campus wurde, wie beschrieben, in 20x20 m große Blöcke zerlegt.

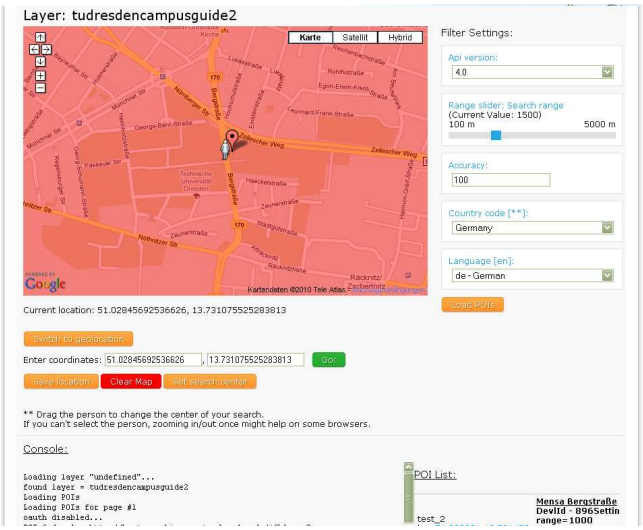


Abb. 5.14: Standort Bergstraße mit zu wenig angezeigten sichtbaren Gebäuden

Im Zentroiden dieses Quadrates wurde der Standort festgelegt. Individuell wurde jeder Standort kontrolliert, ob dieser in einem Gebäude liegt oder auf Vegetation. War dies der Fall, wurde der Standort entweder gelöscht oder innerhalb der Zelle verschoben. Natürlich ist die Wahrscheinlichkeit sehr hoch, dass dabei Fehler aufgetreten sind. So liegen eventuell Standorte immer noch an Stellen, an denen sich ein Baum befindet und deshalb von einer fehlerhaften Höhe ausgegangen werden muss. Zu jedem Standort wird eine Höhe von 1,60 m hinzugerechnet, auf Grund der durchschnittlichen Körpergröße (vgl. Abbschnitt 3.1). Dies ermöglicht die Sicht auf die Gebäude östlich des Trefftzbaus.

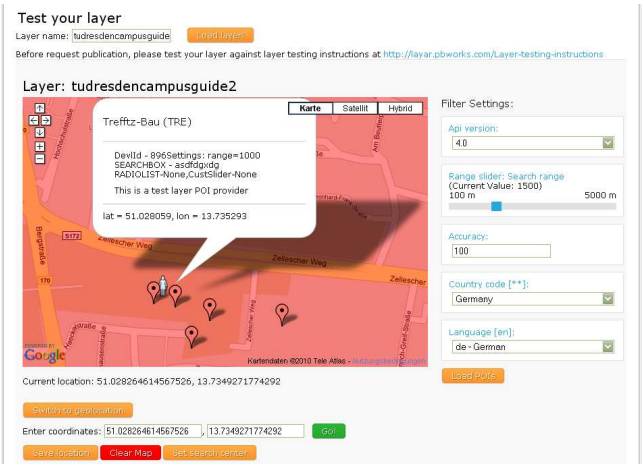


Abb. 5.15: Standort Trefftzbau mit irrtümlich sichtbaren Gebäuden

Diese Probleme treten bei der Nutzung der Testseite von Layar auf. Was passiert nun, wenn man die Anwendung über das Smartphone startet? Das Einwählen ins Internet stellt sich als recht langwierig und kompliziert dar. Dies liegt nicht nur am Gerät, sondern am WLAN der Universität. So muss man sich mit einem Accountnamen und Passwort einloggen. Da die Tasten auf dem Display recht klein sind, vertippt man sich sehr schnell. Die WLAN-Verbindung stellte sich als nicht sonderlich zuverlässig heraus. Der Empfang war an einigen Stellen schwach und ein erneutes Einloggen wurde notwendig, um den Layar-Server ansprechen zu können. Weiterhin musste der GPS-Empfang ausreichend sein. An einigen Stellen auf dem Campus war der dieser allerdings eingeschränkt, bedingt durch die enge Bebauung in diesem Bereich. So musste der Test sogar einmal abgebrochen werden, da die Positionen des Handys in keinster Weise richtig waren. Je länger das Gerät benutzt wurde und der GPS-Empfänger eingeschalten war, desto genauer konnten die Standorte ermittelt werden.
Wurden all diese Hürden genommen, funktioniert die Anwendung recht gut. In vielen Fällen lassen sich die Punkte den Gebäuden eindeutig zuordnen (Abb. 5.16).

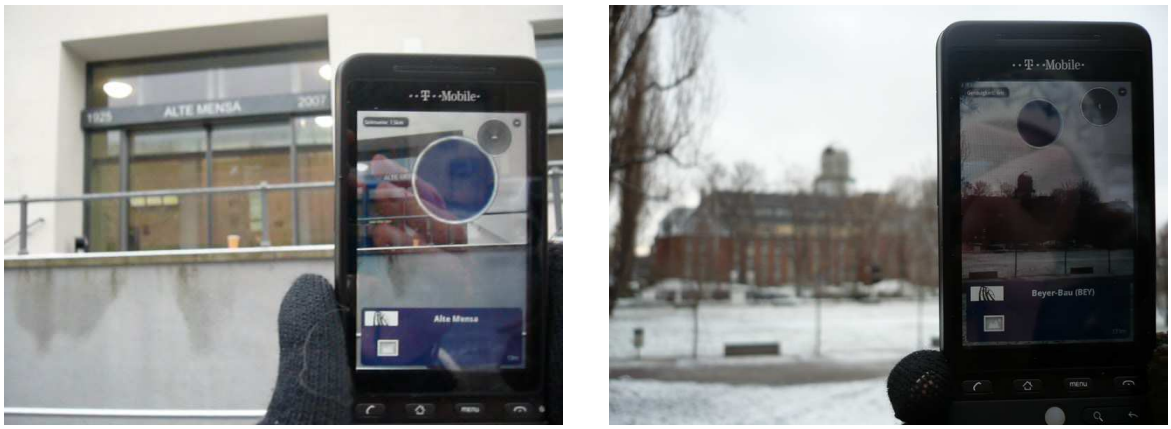


Abb. 5.16: Anzeige des sichtbaren POIs am Beispiel der Alten Mensa und des Beyer-Baus

Gut sichtbar ist der Name des Gebäudes außerdem ist in der unteren rechten Ecke die Entfernung vom Standort bis zu diesem angegeben, was leider in den Abbildungen (5.16) nicht zu erkennen ist. Die Farben wurden entsprechend des TU Dresden Corporate Designs gewählt.

Auch in der Kartenansicht (Abb. 5.17), welche man über die Plattform Layar wählen kann, zeigt sich, dass die Ergebnisse recht gut stimmen. Diese Ansicht ähnelt jener auf der Testseite, wie im vorangegangenen Abschnitt beschrieben wurde.

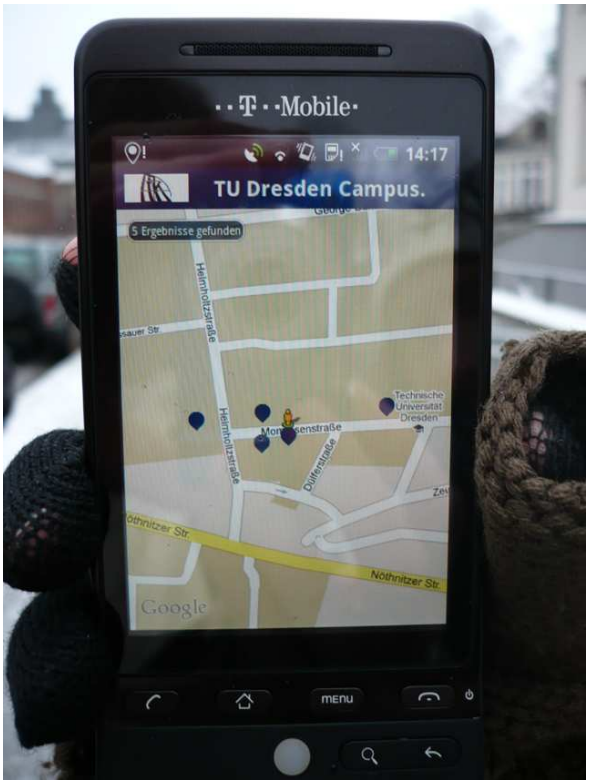


Abb. 5.17: Ansicht der gefunden POIs auf einer Karte

Aber auch bei dem Test des Layers mit dem Smartphone traten größere Fehler auf. So wurde z. B. das Hörsaalzentrum nicht angezeigt (Abb. 5.18), obwohl der Standort des Nutzers einen freien Blick auf dieses Gebäude erlaubt. Der Grund dafür könnte die Ungenauigkeit der Positionsbestimmung sein, sodass die an den Server weitergegebene Position sich doch vor einem Baum oder anderem Hindernis befand.

Layar zeigt eine Genauigkeit der Position von sechs Meter an. Sollte diese Angabe stimmen, könnte es möglich sein, dass der Standort vor dem Baum, welcher rechts im Bild zu sehen ist, genutzt wurde. Somit wäre das Hörsaalzentrum nicht sichtbar.

Des Weiteren bestand bei einigen Punkten das Problem, dass sie zu hoch, zu weit rechts oder links lagen und somit eine Zuordnung zu einem Gebäude nur schwer möglich war. Dies war ein Problem der Genauigkeit. Da bei der Berechnung nur Rasterzellen als Standpunkt gewählt wurden und

der GPS-Empfänger ebenfalls ungenau war, sind die entsprechenden sichtbaren Punkte auch ungenau. Bei einer Weiterentwicklung der Anwendung sollte darauf Wert gelegt werden, eine höhere Genauigkeit zu erzielen. Dies könnte z. B. durch Echtzeitberechnung der Sichtbarkeiten erfolgen.

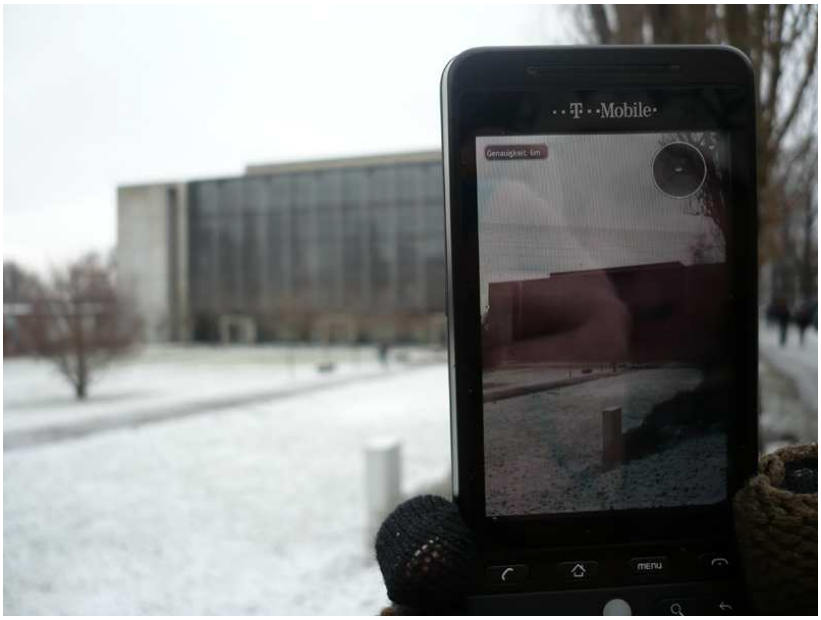


Abb. 5.18: Freier Blick auf das Hörsaalzentrum ohne Anzeige eines POIs

Des Weiteren bestand bei einigen Punkten das Problem, dass sie zu hoch, zu weit rechts oder links lagen und somit eine Zuordnung zu einem Gebäude nur schwer möglich war. Dies ist ein Problem der Genauigkeit. Da bei der Berechnung nur Rasterzellen als Standpunkt gewählt wurden und der GPS-Empfänger ebenfalls ungenau ist, sind die entsprechenden sichtbaren Punkte auch ungenau. Bei einer Weiterentwicklung der Anwendung sollte darauf Wert gelegt werden, eine höhere Genauigkeit zu erzielen. Dies könnte z. B. durch Echtzeitberechnung der Sichtbarkeiten erfolgen.
Getestet wurden neben der Richtigkeit der Punkte auch das Layout (Abb. 5.19). Es wurde unter anderem das Logo der TU Dresden als Icon ausprobiert sowie eine Beschriftung des Hülssebaus mit dessen Kürzel HÜL. Da sich an die Vorschriften von Layar gehalten werden musste, sind diese eigenen Icons allerdings nicht sonderlich ansprechend gestaltet. So war es nicht möglich, den Hintergrund transparent zu schalten, sodass beispielsweise nur die Buchstaben HÜL dargestellt werden. Bei der älteren Version von Layar war eine Transparenz noch möglich, seit ihrer Umstellung aber nicht mehr. Vielleicht erfolgt in Zukunft noch ein weiteres Update, sodass eine Transparenzschal-

tung des Hintergrundes wieder möglich wird und eventuell weitere Gestaltungsmöglichkeiten hinzu kommen.

Das Beispiel des Hülssebaus (Abb. 5.19) zeigt eine Ungenauigkeit der Anwendung. Obwohl das Gebäude nur wenige Meter entfernt ist, wird in der Anwendung eine Distanz von 49 m angegeben. Diese Entfernung kommt zustande, da nicht der Punkt am Gebäudeeingang angezeigt wird, sondern der Schwerpunkt aus den Sichtbarkeitspolygonen (vgl. Kapitel 5.3). Bei einer Verbesserung der Anwendung könnte dieser Aspekt beachtet werden.

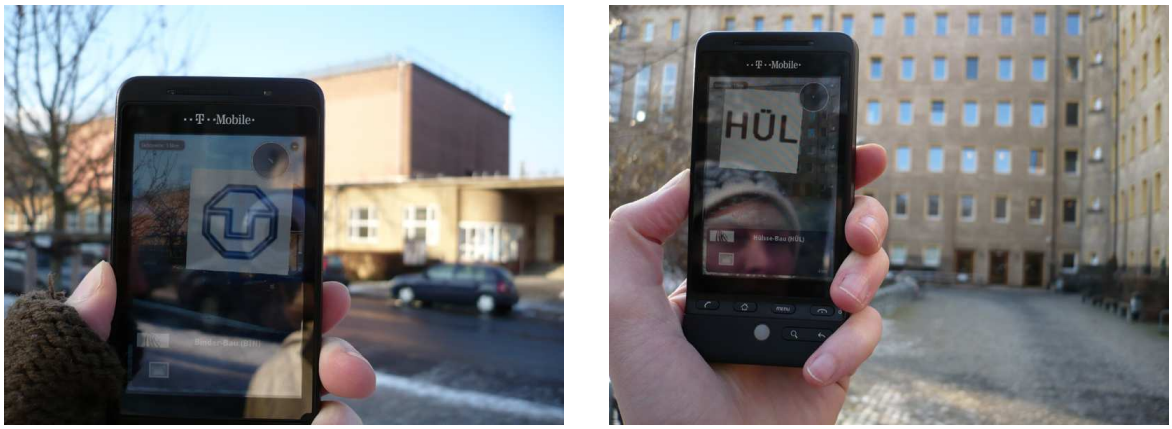


Abb. 5.19: Verschiedene Icons der POIs

5.8 Bewertung der Anwendung

Wie schon an vielen Stellen deutlich wurde, kann die Anwendung in vielerlei Hinsicht noch verbessert werden. Der gesamte Ablauf der Berechnung der Sichtbarkeiten könnte in größerem Umfang automatisch ablaufen, außerdem wären Beschriftungen im Sichtfeld wünschenswert.

Im Idealfall sollte die Anwendung so funktionieren, dass alle Berechnungen nicht vorher durchgeführt werden und die Ergebnisse abgespeichert und aufgerufen werden. Die Vorgänge sollten vielmehr mit Aufruf der Anwendung und Erhalten der aktuellen Position in Gang gesetzt werden. Dazu sind allerdings leistungsfähige Rechner sowie ein größerer Programmieraufwand notwendig. Das Ergebnis sollte dem Nutzer in wenigen Sekunden vorliegen. Dazu müssten alle Berechnung in Echtzeit ablaufen. Denkbar wäre dies durch eine Erweiterung des Java-Codes sowie einer Programmierung in ArcGIS mit Hilfe von ArcObjects. Noch besser wäre eine Programmierung in GRASS GIS, da

hier die Schnittstelle zur Datenbank vorhanden ist. Wird die Berechnung erst mit Aufruf der Anwendung gestartet, ist eine Schleife für die Berechnung aller Punkte nicht mehr notwendig. Demnach wird weniger Speicherplatz benötigt, wodurch auch der Modeler seinen Zweck erfüllen würde, da er nicht auf Grund der Überlastung abbrechen müsste. Allerdings muss dieser vom Nutzer ins ArcGIS geladen und gestartet werden. Durch ein eingebundes Programm wäre dies nicht mehr notwendig. Ob diese Echtzeitberechnung zu realisieren ist, ist eher fragwürdig, da, wie gesagt, die Rechendauer eine große Rolle spielt. Bis das Ergebnis ermittelt ist, müssen, wie beschrieben, viele Arbeitsschritte ablaufen (vgl. Kapitel 5.2, 5.3, 5.5). Länger als zwei bis drei Sekunden sollte der Nutzer nicht auf das Ergebnis warten müssen, da er sonst diese Anwendung nicht nutzen würde, sondern auf konventionelle Methoden der Orientierung (Karte, Lageplan) zurückgreifen würde.

Der gewählte Weg mit einer Vorablage der berechneten Ergebnisse ist nicht besonders genau. Der Standort befindet sich in einer 20x20 m Rasterzelle, wird allerdings nicht genauer bestimmt. Ist die ID der Rasterzelle bekannt, wird die entsprechende Sichtbarkeitsanalyse ausgewählt. Bei einer Genauigkeit des GPS-Empfängers von ca. 20 m ist diese Vorgehensweise durchaus akzeptabel. Doch die Technik entwickelt sich permanent weiter und somit werden sich auch die Technologien zur Positionsbestimmung kontinuierlich weiterentwickeln, was zur Folge hat, dass eine solche Ausdehnung der Rasterzelle nicht mehr ausreichend ist. Demnach sollte, auch unter diesem Gesichtspunkt, eine Berechnung in Echtzeit durchaus angestrebt werden. Die Rechenleistung verbessert sich ebenfalls immer mehr, sodass in naher Zukunft sicherlich ein schnellerer Ablauf der Aufgaben möglich wird.

Wie bereits im vorangegangenem Abschnitt 5.7 deutlich wird, funktioniert die Anwendung auf der Layar-Testseite und dem Smartphone recht gut. Auftretende Fehler sind erklärbar und können dadurch mit Hilfe entsprechender Daten relativ leicht behoben werden. Auf dem Smartphone kommt es zum Teil durch Verbindungsprobleme sowie Verzögerungen bei der Positionsbestimmung zu langen Wartezeiten. Die Störungen der Internetverbindung könnten durch Nutzung von UMTS weitestgehend behoben werden.

6 Fazit und Zusammenfassung

Augmented Reality Anwendungen haben in den letzten Jahren sehr stark an Bedeutung gewonnen. Mit der Verbesserung der mobilen Geräte kommen auch viele Appss auf den Markt, welche das Leben erleichtern oder einfach Spaß machen sollen. Diese Anwendungen stehen zum Teil kostenlos zum Download zur Verfügung, sind aber auch teilweise kostenpflichtig. Nicht nur die Zahl der Anwendungen steigt kontinuierlich, sondern auch die Vielfalt der mobilen Geräte. So entwickeln sich Handys und Laptops immer weiter, aber es gibt auch Geräte, wie Tablet-PCs oder Netbooks, welche zusätzlich auf dem Markt zufinden sind und dank ihrer Anwendungsvielfalt (Nutzung als elektronischer Kalender, mobiles Internet, Kamera usw.) finden all diese Geräte durchaus auch Abnehmer. Tablet-PCs, Netbooks und Laptops lassen ebenfalls die Ausführung bestimmter Apps zu.
Diese Arbeit stellt einen Überblick über mobile Geräte sowie zugehörige AR Anwendungen dar. Aber auch Anwendungen ohne mobile Endgeräte werden vorgestellt. Dabei wird ein besonderes Augenmerk auf den Bereich der Kartographie gelegt.
Weiterhin werden die Grundlagen zu Sichtbarkeitsanalysen dargestellt und wie diese in GIS-Programmen umgesetzt werden. Dabei wird besonders auf die Software ArcGIS eingegangen, da mit Hilfe dieser der praktische Teil der Arbeit bearbeitet wurde. Kapitel 3 enthält außerdem einen Überblick über Möglichkeiten der grafischen Darstellung von dreidimensionalen Objekten auf mobilen Geräten. Da dem Nutzer nur eine eingeschränkte Speicherkapazität zur Verfügung steht, müssen spezielle Anforderungen erfüllt sein, um dreidimensionale Szenen darstellen zu können.
Neben diesen Betrachtungen wird zudem ein Überblick über Beschriftungsmöglichkeiten im dreidimensionalen Raum gegeben. Da die Anwendung Layar, welche bei der praktischen Umsetzung genutzt wurde, nur wenig Spielraum bei der Beschriftung der POIs zulässt, spielt dieser Abschnitt für die Arbeit nur eine untergeordnete Rolle.
Abschließend wurde die praktische Arbeit vorgestellt. Erstellt wurde ein Informationssystem für den Campus der Technischen Universität Dresden. Mit Hilfe der AR Anwendungen lassen sich die Na-

men der Gebäude anzeigen, welche vom Standort des Nutzers sichtbar sind. Alle Gebäude, welche von dieser Position aus nicht zu sehen sind, werden weder durch POIs angezeigt noch mit Namen erwähnt. Dies steht im Gegensatz zu bekannten Anwendungen, welche z. B. alle öffentlichen Gebäude in einem Radius von 5 km anzeigen.

Die Anwendung funktioniert gut, allerdings ist sie an vielen Stellen verbesserungswürdig. Für eine Erweiterung der App ist es vorstellbar, auch dreidimensionale Objekte einfließen zu lassen. So könnten Gebäude, welche sich noch in Bau befinden, schon vor Fertigstellung mit Hilfe eines mobilen Gerätes angezeigt werden. Dadurch könnte man erkennen, wie es sich in die Umgebung einfügt und erhält einen Eindruck vom späteren Aussehen des Objektes. Die Technik entwickelt sich sehr schnell weiter, so wird es bald kein Problem mehr sein, auch komplexe dreidimensionale Objekte auf mobilen Geräten darzustellen. Dies würde den Markt für AR Anwendungen vergrößern und mehr Downloads von Apps hervorrufen.

Auch im Bereich der Fußgänger- und Fahrzeugnavigation entwickeln sich immer bessere Systeme. Der Trend zu höheren Auflösungen bzw. größeren Diplays ist am Markt für Smartphones zu erkennen, sodass diese in Zukunft auch als Navigationsgerät genutzt werden können. Mit der entsprechenden Halterung im Auto erhält der Nutzer ähnlichen Komfort wie mit einem einfachen Navigationsgerät. Auch Projekte mit Hilfe von Augmented Reality zu navigieren, werden immer weiter entwickelt. So kam das Produkt Wikitude Drive [Mobilizy (b)] auf den Markt, welches bei der Fahrzeugnavigation auf AR zurückgreift und das reale Bild mit den virtuellen Inhalten verknüpft, welche zur Wegweisung nowendig sind. Dies hat den Vorteil, dass sich der Nutzer mehr auf den Straßenverkehr konzentrieren kann, da er beim Blick auf das Smartphone die reale Umgebung weiterhin sieht, aber dennoch wichtige zusätzliche Informationen zur Findung des Weges erhält. Dadurch kann er die reale Situation viel besser verfolgen und ist weniger stark abgelenkt.

Diese Entwicklungen zeigen, wie vielversprechend und hilfreich AR Anwendungen sein können, gerade auch im Bereich der mobilen Geräte. Dabei steckt hinter solchen Anwendungen auch ein enormer wirtschaftlicher Wert. So stehen entsprechende Apps z. B teilweise auch als kostenpflichtige Downloads zur Verfügung. Es ist demnach nicht verwunderlich, dass der Apple-Konzern den Download der zehn millardensten App groß gefeiert hat.

Literaturverzeichnis

[Amor 2002] AMOR, Daniel: *Das Handy gegen Zahnschmerzen - und andere Geschäftsmodelle für die Dienstleister von morgen*. Galileo Business, 2002 10

[Apple a] APPLE: *ARSoccer - Augmented Reality Soccer Game*. – URL http://itunes.apple.com/us/app/arsoccer-augmented-reality/id381035151?mt=8. – letzter Abruf: 01.12.2010 - 14, 15

[Apple b] APPLE: *SnapShop Showroom*. – URL http://itunes.apple.com/us/app/snapshop-showroom/id373144101?mt=8. – letzter Abruf: 01.12.2010 - 14, 15

[Bauer 2003] BAUER, Manfred: *Vermessung und Ortung mit Satelliten*. Wichmann Verlag, Heidelberg, 2003 27

[Bauer] BAUER, Martin: *Uni Protokolle - Lexikon - Z-Puffer*. – URL http://www.uni-protokolle.de/Lexikon/Z-Buffer.html. – letzter Abruf: 18.12.2010 - 39

[Bayer 2002] BAYER, Thomas: REST Web Services - Eine Einführung. (2002). – URL http://www.oio.de/public/xml/rest-webservices.htm,http://www.oio.de/public/xml/rest-webservices.pdf. – letzter Abruf: 07.01.2011 - 61

[Blankenbach 2007] BLANKENBACH, Jörg: *Handbuch der mobilen Geoinformation Architektur und Umsetzung mobiler standortbezogener Anwendungen und Dienste unter Berücksichtigung von Interoperabilität*. Herbert Wichmann Verlag Heidelberg, 2007 6, 7, 8, 9

[Blümchen 2002] BLÜMCHEN, Kai: *Augmented Reality*. 2002. – URL http://wwwiaim.ira.uka.de/Teaching/ProseminarMedizin/Ausarbeitungen/WS0102/07_Augmented_Reality.pdf. – Universität Karlsruhe - letzter Abruf: 11.03.2011 - 3

[Brunner 2001] BRUNNER, Kurt ; DRESDEN, TU (Hrsg.): *Kartographische Bausteine - Kartengestaltung für elektronische Bildanzeigen*. 2001 iv, 45, 46, 47

[Brunner 2002] BRUNNER, Kurt: Schlechte Karten. Zur Problematik kartographischer Visualisierung auf kleinformatigen Displays mobiler Endgeräte. In: *GeoBit. Das Magazin für raumbezogene Informationstechnologie* (2002) iv, 45, 46

[Computer-Bild] COMPUTER-BILD: *Größenvergleich Smartphone, Tablet-PC*. – URL http://i.computer-bild.de/imgs/119487595_c247cb1479.jpg. – letzter Abruf: 10.01.2011 - iii, 13

[displaysearch] DISPLAYSEARCH: *New Product Introductions and Announcements Expected to Drive Segmentation of the Slate PC Market*. – URL http://www.displaysearch.com/cps/rde/xchg/displaysearch/hs.xsl/110207_displaysearch_forecasts_200_y_y_growth_for_tablet_pcs.asp. – letzter Abruf: 11.03.2011 - 12

[Dudenredaktion 2006] DUDENREDAKTION (Hrsg.): *Duden - Die deutsche Rechtschreibung*. Bibliographisches Institut & F.A. Brockhaus AG, Mannheim, 2006 10

[EasyTrans] EASYTRANS: *EasyTrans Homerpage*. – URL http://www.geoima.de/download/e_easy.html. – letzter Abruf: 19.08.2010 - 50

[Engels 1998] ENGELS, Ralf: *Raytracing-Verfahren*. 1998. – URL http://theorie.informatik.uni-ulm.de/Lehre/WS9798/Computergrafik/Engels/ausarbeitung.html. – Uni Ulm - letzter Abruf: 05.02.2011 - 40

[ESRI] ESRI: *ArcGIS Desktop Help*. – URL http://help.arcgis.com/de/arcgisdesktop/10.0/help/index.html#//00q90000008n000000.htm. – letzter Abruf: 27.01.2010 - iv, 35, 37, 38

[Firchau 2001] FIRCHAU, Steffen: *Telepointing - Positionsbestimmung und Sichtbarkeitsanalyse durch Kombination terrestrischer Bilder mit 3D-Stadtmodellen*. 2001. – Studienarbeit - Universität Stuttgart, Institut für Photogrammetrie iii, 25

[Fisher 1992] FISHER, Peter: First Experiments in Viewshed Uncertainty: Simulating Fuzzy Viewsheds. In: *American Society for Photogrammetry and Remote Sensing* (1992) 29

[Fisher 1993] FISHER, Peter: Algorithm and implementation uncertainty in viewshed analysis. In: *American Society for Photogrammetry and Remote Sensing* (1993) iii, vii, 29, 30, 31, 32, 33

[Fisher 1996] FISHER, Peter: Extending the Applicability of Viewsheds in Landscape Planning. In: *American Society for Photogrammetry and Remote Sensing* (1996) iii, iv, 29, 33, 34, 35

[Frings 2002] FRINGS, Gabi: *Mobile Computing, Betriebssysteme und Entwicklungsumgebungen (Seminararbeit)*. 2002 7

[Graf 2003] GRAF, Roland J.: Ortsbasierte Sichtbarkeitsanalyse mit digitalen Geländemodellen auf mobilen Endgeräten. (2003), S. 22 38, 39

[Großer 2002] GROSSER, Konrad ; SPEKTRUM, Heidelberg B. (Hrsg.): *Lexikon der Kartographie und Geomatik*. Bollmann, Jürgen and Koch, Wolf Günther, 2002 45, 46

[Hahlen 2004] HAHLEN, Johann: *Statistisches Bundesamt - Pressekonferenz Leben und Arbeiten in Deutschland Ergebnisse des Mikrozensus 2004*. 2004. – URL http://www.destatis.de/jetspeed/portal/cms/Sites/destatis/Internet/DE/Presse/pk/2004/Mikrozensus/Statement__Hahlen,templateId=renderPrint.psml. – Abruf: 23.07.2010 24

[Hennig 1997] HENNIG, Alexander: *Die andere Wirklichkeit*. 1.Auflage. Addison-Wesley Longman Verlag GmbH, Bonn, 1997 4

[Hofmann-Wellenhof u. a. 2001] HOFMANN-WELLENHOF, B. ; LICHTENEGGER, H. ; COLLINS, J.: *Global Positioning System, Theory and Practice*. Springer Verlag, Wien, 2001 27

[IT-Lexikon] IT-LEXIKON: *IT Lexikon - Das große Online-Lexikon für Informationstechnologie*. – URL http://www.itwissen.info/. – letzter Abruf: 13.01.2011 - 11, 12

[IT-Wissen a] IT-WISSEN: *Head Mounted Display*. – URL http://www.itwissen.info/bilder/head-mounted-display-hmd-foto-vr-realities-dot.png. – letzter Abruf: 24.11.2010 - iii, 5

[IT-Wissen b] IT-WISSEN: *Head Up Display*. – URL http://www.itwissen.info/bilder/head-up-display-hud-foto-bmw.png. – letzter Abruf: 24.11.2010 - iii, 5

[Jersey] JERSEY: *Projekt Jersey*. – URL http://jersey.java.net/. – letzter Abruf: 07.01.2011 - 61

[JSON] JSON: *JavaScript Object Notation*. – URL http://www.json.org/. – letzter Abruf: 07.01.2011 - 62

[Karpischek u. a. 2009] KARPISCHEK, Stephan ; HEUEL, Stephan ; MARFORIO, Claudio ; GODENZI, Mike ; MICHAHELLES, Florian: SwissPeaks - Mobile augmented reality to identify mountains. (2009) 17, 18

[Kluge] KLUGE, Mario: *3D - Fußgängernavigation*. – URL http://www.geographie.uni-potsdam.de/component/option,com_portfol/Itemid,625/task,viewcategory/vcatid,295/view,fullview/refid,60/lang,german/. – letzter Abruf: 26.01.2011 - iii, 19

[Kluge 2010] KLUGE, Mario: Fußgängernavigation mit Augmented Reality. In: *26th Chaos Communication Congress*, URL http://events.ccc.de/congress/2009/Fahrplan/attachments/1426_kluge-ccc2009-hq.pdf, 2010. – letzter Abruf: 26.02.2011 - 19

[Kolbe u. a. 2004] KOLBE, Thomas ; SCHULZ, Daniela ; PLÜMER, Lutz: 3D-Beschriftung von Wegevideos für die Fußgängernavigation. In: *Mitteilungen des Bundesamtes für Kartographie und Geodäsie, Band 34, Tagungsband zur Sitzung der Arbeitsgruppe Automation in der Kartographie 2004* (2004) iv, 43, 44

[Kovrig] KOVRIG, Attila: *IcoFX*. – URL http://icofx.ro/. – letzter Abruf: 03.01.2011 - 59

[Kreitlow u. a. 2010] KREITLOW, Stefanie ; BRETTSCHNEIDER, Andrea ; JAHN, Cord-Hinrich ; FELDMANN-WESTENDORFF, Uwe: ETRS/UTM - Der Bezugssystemwechsel und die Auswirkungen auf die Geodatennutzung. In: *Kartographische Nachrichten* 4 (2010) 51

[Lancelle 2003/2004] LANCELLE, Marcel: *Automatische Generierung und Visualisierung von 3D-Stadtmodellen*, Technische Universität Braunschweig Institut für Computergraphik, Diplomarbeit, 2003/2004 40

[Layar a] LAYAR: *Layar - Berliner Mauer*. – URL http://site.layar.com/company/blog/the-berlin-wall-is-back/. – letzter Abruf: 01.12.2010 - iii, 13, 14

[Layar b] LAYAR: *Layar Reality Browser*. – URL http://www.layar.com/. – letzter Abruf: 04.01.2011 - 56, 57

[Layar c] LAYAR: *Layoutgestaltung mit der Plattform Layar*. – URL http://www.layar.com/static/img/tooltips//look_and_feel.png. – letzter Abruf: 06.01.2011 - iv, 58

[Lehmann und Döllner 2010] LEHMANN, Christine ; DÖLLNER, Jürgen: 3D-Beschriftung in interaktiven, virtuellen 3D-Umgebungen durch Verdeckungsminimierung. (2010). – URL http://www.geoinformatik2010.de/public/abstracts/lehmann.pdf iv, 41, 42, 43

[Lindner u. a. 2006] LINDNER, Helmut ; SIMON, Günther ; SIEBKE, Wolfgang ; WUTTKE,

Werner: *Physik für Ingenieure, Band 10*. 17. Auflage. Fachbuchverlag Leipzig im Carl Hanser Verlag, 2006 24

[Maass und Döllner 2006] MAASS, Stefan ; DÖLLNER, Jürgen: Dynamic Annotation of Interactive Environments using Object - Integrated Billboards. In: *14th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision, WSCG, 327-334* (2006) 41

[Mattern 2008] MATTERN, Friedemann ; YORK, Springer-Verlag Berlin Heidelber N. (Hrsg.): *Digitale Visionen - Zur Gestaltung allgegenwärtiger Informationstechnologien*. Roßnagel, Alexander; Sommerlatte, Tom; Winand, Udo, 2008 10

[Merl 2003] MERL, Edgar: Beispiele ortsbasierter Dienste - "Tourist Guide". In: *Ortsbasierte Dienste, Seminar 01919 im Sommersemester 2003, Fachbereich Informatik, Praktische Informatik II, FernUniversität Hagen* (2003) 26

[Milan] MILAN: *Milan Geoservice GmbH*. – URL http://www.milan-flug.de/. – letzter Abruf: 02.01.2011 - 52

[Milgram und Kishino 1994] MILGRAM, Paul ; KISHINO, Fumio: A Taxonomy of Mixed reality Visual Displays. In: *IEICE Transactions on Information Systems, No.12* (1994), S. 15 3

[Mobilizy a] MOBILIZY: *Wikitude*. – URL http://www.wikitude.org/de. – letzter Abruf: 04.01.2011 - 57

[Mobilizy b] MOBILIZY: *Wikitude Drive*. – URL http://www.wikitude.org/de/drive. – letzter Abruf: 10.02.2011 - 76

[Mobilizy c] MOBILIZY: *Wikitude Geo-Tag the World*. – URL http://www.wikitude.me/. – letzter Abruf: 09.01.2011 - 49

[Oracle-Corporation] ORACLE-CORPORATION: *Jersey Wiki*. – URL http://wikis.sun.com/display/Jersey/Main. – letzter Abruf: 07.01.2011 - 61

[OSM] OSM: *Open Street Map - Die freie Wiki-Weltkarte*. – URL http://www.openstreetmap.de/. – letzter Abruf: 11.03.2011 56

[PC-Welt a] PC-WELT: *Neue Smartphones mit Nvidia-Grafik*. – URL http://www.pcwelt.de/news/HTC-Neue-Smartphones-mit-Nvidia-Grafik-46771.html. – letzter Abruf: 05.02.2011 - 38

[PC-Welt b] PC-WELT: *Standortsuche über WLAN*. – URL http://www.pcwelt.de/news/Positionsbestimmung-Standortsuche-ueber-WLAN-konkurriert-mit-GPS-310942.html. – letzter Abruf: 05.02.2011 - 28

[Reichenbacher 2004] REICHENBACHER, Tumasch: *Mobile Cartography - Adaptive Visualisation of Geographic Information on Mobile Devices*. Verlag Dr. Hut, München, 2004. – URL http://tumb1.biblio.tu-muenchen.de/publ/diss/bv/2004/reichenbacher.pdf 26

[Reichwald 2002] REICHWALD, Ralf ; BETRIEBSWIRTSCHAFTLICHER VERLAG DR. TH. GABLER GMBH, Wiesbaden (Hrsg.): *Mobile Kommunikation. Wertschöpfung, Technologien, neue Dienste*. 2002 8, 9

[Reitmayr 2006] REITMAYR, Gerhard: *University of Cambridge - Augmented Maps*. 2006. – URL http://mi.eng.cam.ac.uk/~gr281/augmentedmaps.html. – letzter Abruf: 24.112010 - iii, 16, 17

[Reitmayr und Drummond 2006] REITMAYR, Gerhard ; DRUMMOND, Tom W.: Going out: Robust Model-based Tracking for Outdoor Augmented Reality. (2006) 6

[Reitmayr u. a. 2006] REITMAYR, Gerhard ; EADE, Ethan ; DRUMMOND, Tom: Localisation and Interaction for Augmented Maps, URL http://mi.eng.cam.ac.uk/~gr281/docs/ReitmayrAugmentedMapsPresentation.pdf, 2006. – letzter Abruf: 24.11.2010 - 16, 17

[Roth 2009] ROTH, Michael: *GIS-basierte Sichtraumanalysen als Beitrag zur qualifizierten Prognose von Auswirkungen großer Energieprojekte auf das Landschaftsbild*. Februar 2009. – Technische Universität Dortmund - Fakultät Raumplanung - Lehrstuhl Landschaftsökologie und Landschaftsplanung 23, 24

[Salzburg-research] SALZBURG-RESEARCH: *peak.ar*. – URL http://peakar.salzburgresearch.at/. – letzter Abruf: 26.02.2011 - 18

[Schnabel 2010] SCHNABEL, Patrick: *Elektronik Kompendium - Location Based Services*. 2010. – URL http://www.elektronik-kompendium.de/sites/kom/0905061.htm. – letzter Abruf: 29.07.2010 - 26

[Schulz 2004] SCHULZ, Daniela: *3D-Beschriftung im Objektraum für die videobasierte*

Fußgängernavigation, Rheinische Friedrich-Wilhelms-Universität Bonn, Diplomarbeit, 2004 43, 47

[Tantius 2008] TANTIUS, Richard: Augmented Reality auf mobilen Geräten. (2008). – URL http://sewiki.iai.uni-bonn.de/_media/teaching/seminars/ss08/mobilechi/augmentedreality_2.0.pdf 6, 7

[Tegtmeier 2006] TEGTMEIER, André: *Augmented Reality als Anwendungstechnologie in der Automobilindustrie*, Otto-von-Guericke-Universität Magdeburg, Dissertation, 2006 4, 5

[thehotgadget 2010] THEHOTGADGET: *Größenunterschiede zwischen Smartphone, Netbbook und Laptop*. 2010. – URL http://thehotgadget.com/wp-content/uploads/2010/06/smartphone-vs-laptops1.jpg. – letzter Abruf: 10.01.2011 - iii, 12

[Torge 2003] TORGE, Wolfgang: *Geodäsie*. 2. Auflage. Walter deGryuther GmbH & Co. KG, Berlin, 2003 51

[Trautwein u. a. 2010] TRAUTWEIN, Friedjoff ; FISLER, Joël ; HUGENTOBLER, Marco ; LÜSCHER, Patrick ; WEIBEL, Robert ; HÄGI, Suzette: Geländeanalyse. In: *Geographic Information Technology Training Alliance (GITTA)* (2010) iii, 28, 29

[UMTS] UMTS: *UMTS-Portal*. – URL http://umts.de/. – letzter Abruf: 10.01.2011 - 8

[Universität Heidelberg] UNIVERSITÄT HEIDELBERG, Geographisches I.: *Open Street Map 3D*. – URL http://www.osm-3d.org/home.de.htm. – letzter Abruf: 18.02.2011 - 63

[Wagner 2007] WAGNER, Daniel: *Handheld Augmented Reality*, Technische Universität Graz, Dissertation, 2007. – URL http://studierstube.icg.tu-graz.ac.at/thesis/Wagner_PhDthesis_final.pdf 10

[Wang 2011] WANG, Xuan: *Verschiedene Filter der Plattform Layar*. 2011. – URL http://layar.pbworks.com/w/page/32272647/Fourth%20Layar%20Tutorial%20-%20layer%20with%20filter%20settings. – letzter Abruf: 06.01.2011 - iv, 59, 60

[Wanninger 2005] WANNINGER, Lambert: *Satellitengestützte Positionierung*. 2005. – Vorlesungsskript 27

[Watt 2002] WATT, Alan: *3D-Computergrafik*. Pearson Education Deutschland GmbH, 2002 38

[Wehr und Lohr 1999] WEHR, Aloysius ; LOHR, Uwe: Airborne laser scanning - an introduction and overview. In: *ISPRS Journal of Photogrammetry & Remote Sensing* 54 (1999), S. 68–82 52

[Weibel 2009] WEIBEL, Robert: *Sichtbarkeitsanalyse*. November 2009. – URL http://www.geo.uzh.ch/microsite/geo315/PDF/GEO315_W9_Sichtbarkeit_2010.pdf. – Geographisches Institut - Universität Zürich iii, 28, 29, 30, 31

[Weigel 2007] WEIGEL, Johannes: Sichtbarkeitsanalyse Niedersachsen-Korridor Durchführung einer Ex-Ante-Sichtbarkeitsanalyse mit Hilfe von Visibility Analyst für einen Korridor von der niedersächsischen Küste bis zur Nordrhein-Westfälischen Grenze. (2007), S. 14 23

[Weiss 2002] WEISS, Scott: *Handheld Usability*. John Wiley & Sons, 2002 10

[Wikipedia] WIKIPEDIA: *Wikipedia*. – URL http://wikipedia.de/. – letzter Abruf: 13.01.2011 - 18

[Wildt 2006] WILDT, Steffen: *Mehrwegeausbreitung bei GNSS-gestützter Positionsbestimmung*, Technsiche Universität Dresden, Dissertation, 2006. – URL http://deposit.ddb.de/cgi-bin/dokserv?idn=985865423&dok_var=d1&dok_ext=pdf&filename=985865423.pdf 52

[Zürich] ZÜRICH, ETH: *SwissPeaks*. – URL http://peaks-app.ch/. – letzter Abruf: 26.11.2010 - iii, 18

Anhang A

Workflow

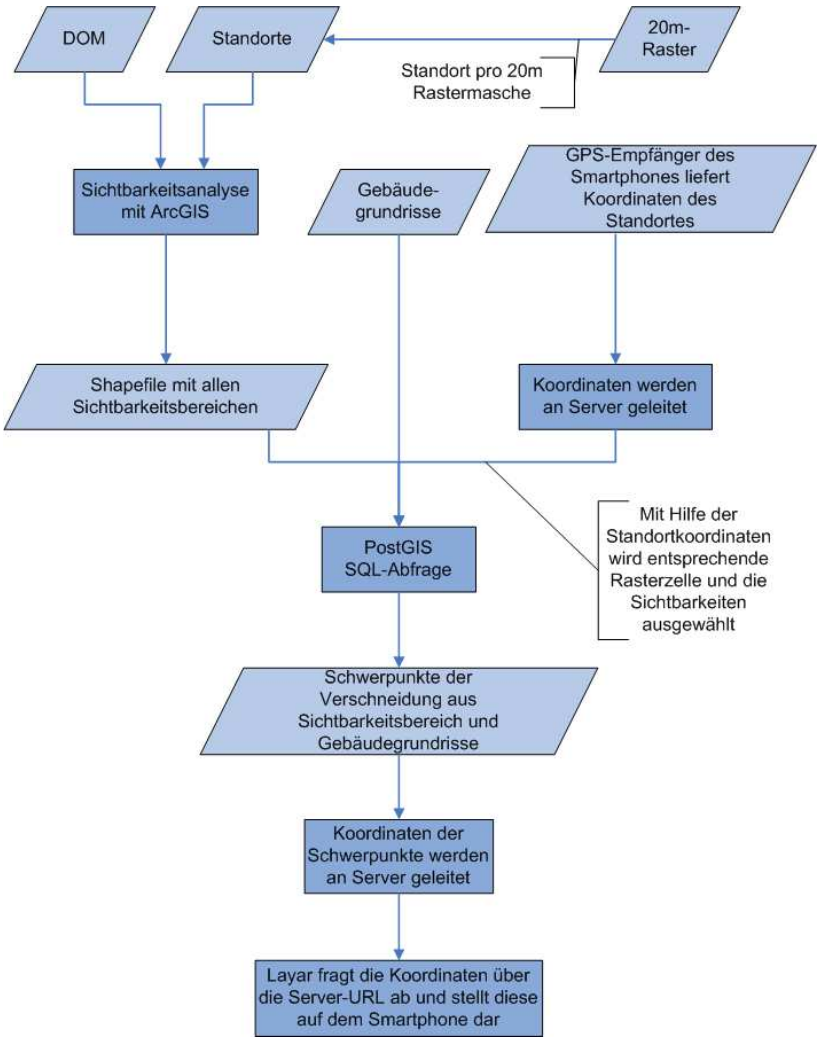


Abb. A.1: Digitaler Workflow der Arbeitschritte zur Erstellung des Campusführers

Anhang B

Quellcode

B.1 PointsOfInterest.java

```
 package de.tud.ifk.ifklayar2;

import java.sql.Connection;
import java.sql.ResultSet;

import javax.ws.rs.DefaultValue;
import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.Produces;
import javax.ws.rs.QueryParam;
import javax.ws.rs.core.Response;

import de.tud.ifk.tools.PostGisTools;
import de.tud.ifk.tools.Tools;
import de.tud.ifk.tools.app.DBApplication;
import de.tud.ifk.tools.jdbc.DriverUtilities;
import de.tud.ifk.tools.jdbc.JDBCManager;

@Path("/getPointsOfInterest")
public class GetPointsOfInterest extends DBApplication {

    // TODO: read documentation
    // https://jersey.dev.java.net/nonav/documentation/latest/user-guide.html
```

```
// http://grass.itc.it/gdp/html_grass64/r.los.html
// http://grass.osgeo.org/wiki/GRASS_AddOns#r.viewshed
// http://www.ucl.ac.uk/~tcrnmar/GIS/r.cva.html

    // Testrequest:
    // http://kartographie.geo.tu-dresden.de/ifklayar2/getPointsOfInterest/?
        lang=DE&countryCode=DE&lon=13.7237691879&userId=5
        a830bc13ee616f116a1d79f27780501a3ec0012&developerId=2881&
        developerHash=9aa43059c2f60b20f2e068814cbc66d09270c5e6&version=3.0&
        radius=1500&timestamp=1283153877210&lat=51.0285383463&layerName=
        tudresdencampus2&accuracy=64

@GET
@Produces("text/plain")
public Response getPointsOfInterest(@DefaultValue("default value")
    @QueryParam("lat") String lat, @QueryParam("lon") String lon) {

    // if (false) {
            // String time = "text/plain: IfK Layar Service; Time: " + new
                Date(System.currentTimeMillis()).toString() + "; query param
                test: " + test;
            // return Response.ok(getLayarDummyData()).build();
            // return Response.ok(getGardeyaDummyData()).build();
            // return Response.ok(time).build();
    // }

    Connection connection = null;

    try {

            StringBuffer debug = new StringBuffer(512);

            // get (pooled) database connection
            JDBCManager.addPool(DriverUtilities.POSTGRES, DriverUtilities.
                HOST, DriverUtilities.DB_NAME, DriverUtilities.USER_NAME,
                DriverUtilities.PASSWORD, maxConnections(),
                initialConnections());
```

```
connection = JDBCManager.getConnection(DriverUtilities.POSTGRES);

// query database
        ResultSet resultSet = PostGisTools.executeQuery(
            connection, IfKLayarQueryBuilder.getQuery(Tools.
            getDouble(lat, 0), Tools.getDouble(lon, 0)), null,
            debug);

        // The String to be built.
        StringBuffer sb = new StringBuffer(512);

        if (resultSet == null) {
                sb.append(debug.toString());
        }
        else {
                sb.append("{\n");
                // the layer name need to be identical as in http
                    ://www.layar.com/publishing
        sb.append("\t\"layer\":␣\"tudresdencampusguide2\",\n");
        sb.append("\t\"errorCode\":␣0,\n");
        sb.append("\t\"errorString\":␣\"ok\",\n");
        sb.append("\t\"hotspots\":\n");
        sb.append("\t[\n");

        int numberOfPoints = 0;

        // the hotspots from the database query
        while (resultSet.next()) {
                sb.append("\t\t{\n");
                sb.append("\t\t\t\"id\":␣\""+resultSet.getInt("
                    osm_id")+"\",\n␣");
                sb.append("\t\t\t\"distance\":␣"+resultSet.
                    getDouble("distance")+",\n␣");
                sb.append("\t\t\t\"title\":␣\""+resultSet.
                    getString("name")+"\",\n␣");
                sb.append("\t\t\t\"type\":␣1,\n␣");
                sb.append("\t\t\t\"lat\":␣"+Math.round(resultSet.
                    getDouble("y")*1000000)+",\n␣");
```

```
                                sb.append("\t\t\t\"lon\":␣"+Math.round(resultSet.
                                    getDouble("x")*1000000)+",\n␣");
                                sb.append("\t\t\t\"actions\":␣[]\n");
                                sb.append("\t\t}\n");
                                        sb.append("\t\t,\n");
                                        numberOfPoints++;
                                }
                        if (numberOfPoints > 0) {
                                // delete very last comma to get valid json
                                    output
                                sb.deleteCharAt(sb.length()-1);
                                sb.deleteCharAt(sb.length()-1);
                        }
                        sb.append("\n");
                                sb.append("\t]\n");
                        sb.append("}\n");
                        }

                        return Response.ok(sb.toString()).build();
        }
        catch (Exception e) {
                e.printStackTrace();
                return Response.ok(Tools.getStackTrace(e)).build();
        }
        finally {
                    if (connection != null) {
                        JDBCManager.free(connection, DriverUtilities.POSTGRES);
                    }
                }
    }

    public String getLayarDummyData() {

        String layar = "{\n" +
        "\"hotspots\":␣[\n" +

        "{\"distance\":␣1000,␣\"attribution\":␣\"This␣is␣a␣test␣layer␣POI␣
            provider\",␣\"title\":␣\"My␣layer␣-␣snowy4,␣location␣=␣51.029111,␣
```

```
            13.723111\",␣\"lat\":␣51029111,␣\"lon\":␣13723111,␣\"imageURL\":␣null
            ,␣\"line4\":␣\"RADIOLIST-None,CustSlider-None\",␣\"line3\":␣\"
            SEARCHBOX␣-␣asdfdgxdg\",␣\"line2\":␣\"DevlId␣-␣896Settings:␣range
            =1000\",␣\"actions\":␣[],␣\"type\":␣0,␣\"id\":␣\"test_1\"},\n" +
        "{\"distance\":␣1000,␣\"attribution\":␣\"This␣is␣a␣test␣layer␣POI␣
            provider\",␣\"title\":␣\"My␣layer␣-␣snowy4,␣location␣=␣51.029222,␣
            13.723222\",␣\"lat\":␣51029222,␣\"lon\":␣13723222,␣\"imageURL\":␣null
            ,␣\"line4\":␣\"RADIOLIST-None,CustSlider-None\",␣\"line3\":␣\"
            SEARCHBOX␣-␣asdfdgxdg\",␣\"line2\":␣\"DevlId␣-␣896Settings:␣range
            =1000\",␣\"actions\":␣[],␣\"type\":␣0,␣\"id\":␣\"test_2\"},\n" +
        "{\"distance\":␣1000,␣\"attribution\":␣\"This␣is␣a␣test␣layer␣POI␣
            provider\",␣\"title\":␣\"My␣layer␣-␣snowy4,␣location␣=␣51.029333,␣
            13.723333\",␣\"lat\":␣51029333,␣\"lon\":␣13723333,␣\"imageURL\":␣null
            ,␣\"line4\":␣\"RADIOLIST-None,CustSlider-None\",␣\"line3\":␣\"
            SEARCHBOX␣-␣asdfdgxdg\",␣\"line2\":␣\"DevlId␣-␣896Settings:␣range
            =1000\",␣\"actions\":␣[],␣\"type\":␣0,␣\"id\":␣\"test_3\"},\n" +
        "{\"distance\":␣1000,␣\"attribution\":␣\"This␣is␣a␣test␣layer␣POI␣
            provider\",␣\"title\":␣\"My␣layer␣-␣snowy4,␣location␣=␣51.029444,␣
            13.723444\",␣\"lat\":␣51029444,␣\"lon\":␣13723444,␣\"imageURL\":␣null
            ,␣\"line4\":␣\"RADIOLIST-None,CustSlider-None\",␣\"line3\":␣\"
            SEARCHBOX␣-␣asdfdgxdg\",␣\"line2\":␣\"DevlId␣-␣896Settings:␣range
            =1000\",␣\"actions\":␣[],␣\"type\":␣0,␣\"id\":␣\"test_4\"}\n" +

        "],\n" +
        "\"layer\":␣\"snowy4\",\n" +
        "\"errorString\":␣\"ok\",\n" +
        "\"morePages\":␣false,\n" +
        "\"errorCode\":␣0,\n" +
        "\"nextPageKey\":␣null\n" +
        "}\n";

        return layar;
    }

    public String getGardeyaDummyData() {

        // test webclient: http://www.gardeya.de/layar/httpdocs/webclient/
        // webservice url: http://141.30.75.5/layar/getPointsOfInterest
```

```
// webservice url: (future) http://kartographie.geo.tu-dresden.de/layar/
    getPointsOfInterest
String gardeya = "{\n" +
                 "\"nextPageKey\":\"2\",\n" +
                 "\"morePages\":true,\n" +
                 "\"hotspots\":[\n" +
                 "{\"actions\":[{\"uri\":\"http://kartographie.geo.tu-
                     dresden.de\",\"label\":\"Zeige␣Webseite\"}],\"
                     attribution\":\"Quelle:␣Competenzatlas␣IT\",\"
                     distance\":3647,\"id\":\"2252\",\"imageURL\":\"http
                     ://kartographie.geo.tu-dresden.de/imgLogos/logoIfK.
                     gif\",\"lat\":51031111,\"lon\":13721111,\"line2\":\"
                     Schlosserstra\u00dfe␣2\",\"line3\":\"70180␣Stuttgart
                     \",\"line4\":\"Distance:␣%distance%\",\"title\":\"
                     Hohe␣Stra?e\",\"type\":0},\n" +
                 "{\"actions\":[{\"uri\":\"http://tu-dresden.de\",\"label
                     \":\"Zeige␣Webseite\"}],\"attribution\":\"Quelle:␣
                     Competenzatlas␣IT\",\"distance\":4834,\"id
                     \":\"2223\",\"imageURL\":\"http://kartographie.geo.tu
                     -dresden.de/imgLogos/logoIfK.gif\",\"lat
                     \":51032222,\"lon\":13722222,\"line2\":\"B\
                     u00f6blinger␣Stra\u00dfe␣32\",\"line3\":\"70178␣
                     Stuttgart\",\"line4\":\"Distance:␣%distance%\",\"
                     title\":\"Schumannbau\",\"type\":0},\n" +
                 "{\"actions\":[{\"uri\":\"http://tu-dresden.de\",\"label
                     \":\"Zeige␣Webseite\"}],\"attribution\":\"Quelle:␣
                     Competenzatlas␣IT\",\"distance\":2210,\"id
                     \":\"2220\",\"imageURL\":\"http://kartographie.geo.tu
                     -dresden.de/imgLogos/logoIfK.gif\",\"lat
                     \":51033333,\"lon\":13723333,\"line2\":\"Junghansstr
                     .5\",\"line3\":\"70469␣Stuttgart\",\"line4\":\"
                     Distance:␣%distance%\",\"title\":\"M?nchner␣Stra?e
                     \",\"type\":0},\n" +
                 "{\"actions\":[{\"uri\":\"http://tu-dresden.de\",\"label
                     \":\"Zeige␣Webseite\"}],\"attribution\":\"Quelle:␣
                     Competenzatlas␣IT\",\"distance\":2921,\"id
                     \":\"2217\",\"imageURL\":\"http://kartographie.geo.tu
                     -dresden.de/imgLogos/logoIfK.gif\",\"lat
```

```
                        \":51034444,\"lon\":13724444,\"line2\":\"Theodor-
                            Heuss-Str.␣2A\",\"line3\":\"70174␣Stuttgart\",\"line4
                            \":\"Distance:␣%distance%\",\"title\":\"M?nchner␣
                            Platz\",\"type\":0},\n" +
                        "{\"actions\":[{\"uri\":\"http://tu-dresden.de\",\"label
                            \":\"Zeige␣Webseite\"}],\"attribution\":\"Quelle:␣
                            Competenzatlas␣IT\",\"distance\":2719,\"id
                            \":\"2214\",\"imageURL\":\"http://kartographie.geo.tu
                            -dresden.de/imgLogos/logoIfK.gif\",\"lat
                            \":51035555,\"lon\":13725555,\"line2\":\"Filmhaus␣
                            Friedrichstr.␣23A\",\"line3\":\"70173␣Stuttgart\",\"
                            line4\":\"Distance:␣%distance%\",\"title\":\"B\
                            u00fcro\",\"type\":0},\n" +
                        "{\"actions\":[{\"uri\":\"http://kartographie.geo.tu-
                            dresden.de\",\"label\":\"Zeige␣Webseite\"}],\"
                            attribution\":\"Quelle:␣Competenzatlas␣IT\",\"
                            distance\":3548,\"id\":\"2173\",\"imageURL\":\"http
                            ://kartographie.geo.tu-dresden.de/imgLogos/logoIfK.
                            gif\",\"lat\":51036666,\"lon\":13726666,\"line2\":\"T
                            \u00fcbinger␣Str.␣7\",\"line3\":\"70178␣Stuttgart
                            \",\"line4\":\"Distance:␣%distance%\",\"title\":\"D?
                            ner\",\"type\":0}\n" +
                        "]\n," +
                        "\"layer\":\"tudresdencampusguide\"\n," +
                        "\"errorCode\":0,\n" +
                        "\"errorString\":\"\"}\n";

        return gardeya;
    }
}
```

B.2 IfKLayarQueryBuilder.java

```
package de.tud.ifk.ifklayar2;


public class IfKLayarQueryBuilder {

        public static String getQuery(double latitude, double longitude) {

                StringBuffer sb = new StringBuffer(512);

        // sub-select
        sb.append("SELECT␣c.name␣AS␣name,␣c.osm_id␣AS␣osm_id,␣st_x((st_centroid(
            st_collect(st_intersection(v.the_geom,␣c.the_geom)))))␣AS␣x,␣st_y((
            st_centroid(st_collect(st_intersection(v.the_geom,␣c.the_geom)))))␣AS
            ␣y,");
        sb.append("round(st_distance(st_transform(setsrid(st_geomfromtext('POINT(
            "+longitude+"␣"+latitude+")'),␣4326),␣32633),␣st_transform(setsrid(c.
            the_geom,␣4326),␣32633)))␣AS␣distance␣");
        sb.append("FROM␣");
        sb.append("layar_viewsheds␣AS␣v,␣layar_raster␣AS␣r,␣layar_campus␣AS␣c␣");
        sb.append("WHERE␣");
        sb.append("st_contains(r.the_geom,␣st_geomfromtext('POINT("+longitude+"␣"
            +latitude+")'))␣AND␣r.\"ObserverID\"␣=␣v.\"ObserverID\"␣␣AND␣
            st_intersects(v.the_geom,␣c.the_geom)␣");
        sb.append("GROUP␣BY␣c.name,␣c.osm_id,␣c.the_geom;␣");



        return sb.toString();
        }

        public static String getQueryOsmId(double latitude, double longitude) {

                StringBuffer sb = new StringBuffer(512);

        // sub-select
        sb.append("SELECT␣c.osm_id␣AS␣osm_id␣FROM␣");
```

```
        sb.append("layar_viewsheds␣AS␣v,␣layar_raster␣AS␣r,␣layar_campus␣AS␣c␣");
        sb.append("WHERE␣");
        sb.append("st_contains(r.the_geom,␣st_geomfromtext('POINT("+longitude+"␣"
            +latitude+")'))␣AND␣r.\"ObserverID\"␣=␣v.\"ObserverID\"␣␣AND␣
            st_intersects(v.the_geom,␣c.the_geom)␣");
        sb.append("GROUP␣BY␣c.osm_id");

        return sb.toString();
        }

        public static String getQueryCentroid(int osmId, double latitude, double
            longitude) {

                StringBuffer sb = new StringBuffer(512);

        // sub-select
        sb.append("SELECT␣c.name␣AS␣name,␣st_x((st_centroid(st_collect(c.the_geom
            ))))␣AS␣x,␣st_y((st_centroid(st_collect(c.the_geom))))␣AS␣y␣");
        sb.append("FROM␣layar_viewsheds␣AS␣v,␣layar_campus␣AS␣c,␣layar_raster␣AS␣
            r␣");
        sb.append("WHERE␣");
        sb.append("c.osm_id␣=␣"+osmId+"␣AND␣st_intersects(v.the_geom,␣c.the_geom)
            ␣AND␣st_contains(r.the_geom,␣st_geomfromtext('POINT("+longitude+"␣"+
            latitude+")'))␣AND␣r.\"ObserverID\"␣=␣v.\"ObserverID\"␣");
        sb.append("GROUP␣BY␣c.name␣");

        return sb.toString();
        }


        // -- SELECT astext(st_centroid(st_collect(c.the_geom))) FROM
            layar_viewsheds AS v, layar_campus AS c WHERE c.osm_id = 23326177 AND
             st_intersects(v.the_geom, c.the_geom) AND v."ObserverID" = 0

}
```

B.3 pom.xml

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣http://maven.apache.org/maven-v4_0_0.xsd">


        <modelVersion>4.0.0</modelVersion>

        <groupId>de.tud.ifk</groupId>
        <artifactId>ifklayar2</artifactId>
        <version>0.0.1-SNAPSHOT</version>
        <packaging>war</packaging>
        <name>ifklayar2</name>
        <url>http://maven.apache.org</url>

        <properties>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        </properties>

        <dependencies>
                <dependency>
                        <groupId>junit</groupId>
                        <artifactId>junit</artifactId>
                        <version>3.8.1</version>
                        <scope>test</scope>
                </dependency>
                <dependency>
                        <groupId>com.sun.jersey</groupId>
                        <artifactId>jersey-server</artifactId>
                        <version>1.1.5-ea-SNAPSHOT</version>
                </dependency>
                <dependency>
                        <groupId>org.json</groupId>
                        <artifactId>json</artifactId>
                        <version>20090211</version>
```

```
            </dependency>
            <dependency>
                    <groupId>org.postgis</groupId>
                    <artifactId>postgis-jdbc</artifactId>
                    <version>1.3.3</version>
            </dependency>
            <dependency>
                    <groupId>postgresql</groupId>
                    <artifactId>postgresql</artifactId>
                    <version>8.4-701.jdbc4</version>
            </dependency>
            <dependency>
                    <groupId>de.tud.ifk</groupId>
                    <artifactId>ifkTools2</artifactId>
                    <version>0.0.1-SNAPSHOT</version>
            </dependency>
    </dependencies>

    <repositories>
            <repository>
                    <id>maven2-repository.dev.java.net1</id>
                    <name>Java.net repository</name>
                    <url>http://repo1.maven.org/maven</url>
            </repository>
            <repository>
            <id>maven2-repository.dev.java.net</id>
            <name>Java.net Maven 2 Repository</name>
            <url>http://download.java.net/maven/2</url>
    </repository>
            <repository>
                    <id>osgeo</id>
                    <name>Open Source Geospatial Foundation Repository</name>
                    <url>http://repo.opengeo.org/</url>
            </repository>
            <repository>
                    <id>java.net</id>
                    <url>http://download.java.net/maven/1</url>
            </repository>
```

```
        <repository>
                <id>svn-ifk</id>
                <url>http://kartographie.geo.tu-dresden.de/svn/ifk/</url>
        </repository>
        <repository>
                <id>osgeo-repo</id>
                <url>http://download.osgeo.org/webdav/geotools</url>
        </repository>
</repositories>

<!-- set scm -->
<!--<scm>
        <developerConnection>
                scm:svn:https://svnhost.net/svnroot/trunk/new-app
        </developerConnection>
</scm>-->

<build>
        <finalName>ifklayar2</finalName>
        <plugins>
                <!-- maven tomcat plugin: http://mojo.codehaus.org/tomcat
                    -maven-plugin/usage.html -->
                <plugin>
                        <groupId>org.codehaus.mojo</groupId>
                        <artifactId>tomcat-maven-plugin</artifactId>
                        <version>1.0</version>
                        <configuration>
                                <url>http://kartographie.geo.tu-dresden.
                                    de/manager/html</url>
                                <server>webserver</server>
                                <username>tomcat</username>
                        <password>vigkib</password>
                        </configuration>
                </plugin>

                <plugin>
                        <groupId>org.apache.maven.plugins</groupId>
                        <artifactId>maven-resources-plugin</artifactId>
```

```
                        <version>2.4.3</version>
                        <configuration>
                                <encoding>UTF-8</encoding>
                        </configuration>
                </plugin>

                <plugin>
                        <groupId>org.codehaus.cargo</groupId>
                        <artifactId>cargo-maven2-plugin</artifactId>
                        <version>1.0</version>
                </plugin>

                <!-- Setting Compiler Version -->
                <plugin>
                        <artifactId>maven-compiler-plugin</artifactId>
                        <configuration>
                                <source>1.5</source>
                                <target>1.5</target>
                        </configuration>
                </plugin>

                <!-- repository / scm -->
                <!--<plugin>
                        <artifactId>maven-release-plugin</artifactId>
                        <configuration>
                                <tagBase>
                                        https://svnhost.net/svnroot/tags
                                </tagBase>
                        </configuration>
                </plugin>-->
        </plugins>
    </build>
</project>
```